UNIVERSITY OF GENOA

PHD PROGRAM IN BIOENGINEERING AND ROBOTICS

On the role of gestures in human-robot interaction

by

Alessandro Carfì

Thesis submitted for the degree of *Doctor of Philosophy* (32° cycle)

May 2020

Fulvio Mastrogiovanni — Supervisor

Giorgio Cannata — Head of the PhD program

Thesis Reviewers:

Toshiyuki Murakami, *Keio University*

Diego Resende Faria, *Aston University*

Thesis Jury:

Matej Hoffman, *Czech Technical University* — External examiner

Diego Resende Faria, *Aston University* — External examiner

Antonio Sgorbissa, *University of Genoa* — Internal examiner

Dibris

Department of Informatics, Bioengineering, Robotics and Systems Engineering

To my grandmother Irene Lancini, I miss you.

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Alessandro Carfì
May 2020

Acknowledgements

My deep gratitude goes first to Professor Fulvio Mastrogiovanni for his precious supervision, encouragement and advice. I am grateful for all the opportunities for personal growth that you gave me in these years.

I would like to thank Barbara Bruno for all the support she gave me. You are a good friend and the kind of researcher that I hope to become. Thanks to all my colleagues that made my journey more pleasant in particular Roberto Menicatti, Luca Buoncompagni, Syed Yusha Kareem, Alessandro Albini and Kourosh Darvish.

I am grateful to all the friends that stood by me in this long adventure. Special thanks to Marina Lemucchi, for the amazing drawings included in this thesis, and to Luca Mantelli, Christopher Benassi and Lorenzo Burrone, for their friendship.

This thesis would not have been possible without the support and the financial assistance provided by my family, my mother Simonetta, my father Mauro and my brother Iacopo.

Finally, I owe my deepest gratitude to Giulia Stasi for her patience, encouragement and continuous support throughout my years of study and through the process of researching and writing this thesis. I love you!

Abstract

This thesis investigates the gestural interaction problem and in particular the usage of gestures for human-robot interaction. The lack of a clear definition of the problem statement and a common terminology resulted in a fragmented field of research where building upon prior work is rare. The scope of the research presented in this thesis, therefore, consists in laying the foundation to help the community to build a more homogeneous research field.

The main contributions of this thesis are twofold: (i) a taxonomy to define gestures; and (ii) an ingegneristic definition of the gestural interaction problem. The contributions resulted is a schema to represent the existing literature in a more organic way, helping future researchers to identify existing technologies and applications, also thanks to an extensive literature review.

Furthermore, the defined problem has been studied in two of its specialization: (i) direct control and (ii) teaching of a robotic manipulator, which leads to the development of technological solutions for gesture sensing, detection and classification, which can possibly be applied to other contexts.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Human Robot Interaction in Industrial Scenario

Manufacturing industries are expected to face a decisive paradigm shift, according to the German federal government Industry 4.0 program, which envisions a tight relationship between good producers and customers, and therefore highly dynamic factories, as the future of automation [1]. A closer relationship with customers and a fast market evolution will likely require high standards of flexibility. Smart factories tackle these requirements by adapting and modifying the production and supply chains to evolving market needs [2]. In order to make this vision possible, humans are expected to work alongside and cooperate with robots, which will have to be often reprogrammed to adapt to new tasks. The Industry 4.0 paradigm requires techniques for fast and easy-to-attain robot task reconfiguration and for enhancing the human-robot interaction. The need of a shared working environment in which humans and robots could cooperate in a safe and productive way had drove the research in different fields. Safety in human robot interaction have been achieved using torque sensors [3], touch sensors [4] and vision [5]. At the same time new solutions, to schedule the work between human operators and robots [6] and to program robots, [7] have been proposed. Moreover, new alternatives have been explored to enrich the human-robot interaction leveraging speech [8], touch screens [9] and gestures.

For our work we have focused on the gestural interaction and its possible application in an industrial scenario.

1.2 Gesture Based Human Robot Interaction

Gestures have been explored as a communication channel both in human-computer (HCI) and human-robot (HRI) interaction. Usually, gestures are aimed at unlocking a new communication channel to send commands to a machine. We refer to this kind of interaction as *functional* human-machine interaction (HMI), and we distinguish it from the notion of *social* HMI[1], according to which a machine encodes and exploits human cognitive models for better interaction.

In this thesis, we want to critically analyse the widely accepted idea that gesture-based interaction is a more natural alternative to classic tools such as the keyboard, mouse or joystick. Gesture-based interaction is of course *natural* for social HMI, as well as in functional HMI scenarios in which each human motion can be piece-wise mapped to a given machine status, e.g., the manipulation of objects in virtual reality by tracking the human hand, where the virtual object pose can be mapped to a Cartesian reference frame centred on the hand itself. Nevertheless, when we define a synthetic gestural language to substitute such tools as the keyboard and the mouse, we are not creating a natural interaction modality [10], since we are not defining a new interaction language, rather we superimpose the gestural language to the one entailed by the keyboard or mouse. To this extent, gesture-based interaction provides an alternative that in certain scenarios can yield a better user experience. For example, in automotive, gestures can be used to interact with the infotainment system without taking the eyes off the road [11]. Therefore, the overall idea in developing gesture-based interfaces should be to provide a new easy-to-use tool able to substitute or at least integrate the classic ones.

In the literature, gesture-based interaction with machines has gained much attention in the past few years in the context of Industry 4.0, whereby inherently safe, task-adaptive, and easy-to-program collaborative robots are expected to work alongside and cooperate with human operators in shop-floor or warehouse environments [1]. The need for a shared workspace where human operators and robots can perform turn-taking or joint operations safely and effectively has steered research in human-robot collaboration (HRC) along different directions. Whilst safety aspects in HRC have been predominant in research, and have been grounded by the use of different sensing modalities, e.g., force/torque sensors [3], touch sensors [23][4] and vision [5], also issues related to human-robot task allocation [6][24], and robot behaviour programming [7] have been investigated. New alternatives have been

[1] For the analysis considered in this paper, we argue that there are many common aspects for what regards HCI and HRI, and therefore we will refer in the paper to the broader notion of HMI.

Table 1.1 Summary of gestural usage for human-robot interaction in the literature.

Article	*Sensor*	*Number of gestures*	*Description*
[12]	Stereo camera	6	Static arm poses
[13]	Camera	-	Discrete hand and fingers motions
[14]	RGB camera	8	Bi-manual static hand and fingers poses
[15]	Accelerometer	12	Discrete arm motions
[16]	Accelerometer	6	Discrete arm motions
[17]	RGB-D camera	5	Discrete arm motions
[18]	RGB-D camera	10	Discrete arm motions
[19]	RGB-D camera	1	Static hand pose
[20]	Accelerometer	-	Continuous arm motion
[21]	RGB-D camera	-	Continuous hand motion
[22]	RGB camera	8	Static hand and fingers poses

explored as well to enrich the interaction process, leveraging speech [8], touch screens [9] and human gestures.

Human-robot gestural interaction has been explored since late 1990s when the recognition of six different arm gestures has been used to control a wheeled robot [12]. Gesture-based interaction has been traditionally paired with speech-based interaction [25, 26] to enrich the communication spectrum, and it can substitute speech entirely in noisy environments [12], or used to substitute such tools as teach pendants when human operators can not get their hands free. Since first attempts, gesture recognition has been used in different applications to interact with robots. User-defined hand and finger gestures have been coupled together with face identification to allow people with disabilities to control an intelligent wheelchair [13]. Similarly, head orientation and bi-manual gestures perceived using an RGB camera have been selected to communicate with the pet-like robot AIBO[2] [14]. Alternatively, wrist-worn accelerometers have been used to detect arm gestures aimed at controlling a 6-DoF manipulator [15], and providing commands to wheeled robots [16]. Arm gestures

[2]Web: https://us.aibo.com/

have been explored to provide task level information to a mobile manipulator [17], and to control a wheeled robot [18]. While pointing gestures have proved useful in indicating directions to assist a humanoid robot in navigation tasks [19], the usage of gestures to specify commands has been explored in difficult scenarios where other kinds of communication are very likely to fail, such as underwater [22]. The communication of a discrete command to an intelligent system implies the usage of a *discrete* gesture, while continuous commands used to tele-operate a wheeled robot [20] or a manipulator [21] necessitate *continuous* gestures[3].

Table 1.1 summarises the previously discussed papers, describing the used sensors, the number of gestures (if applicable), and the gesture types. In the Table, we could specify the number of gestures when the referenced paper was considering discrete gestures and not continuous ones since discrete gestures imply the presence of a gesture *dictionary*. Although HRC is only one of the possible examples whereby gesture-based interaction can be exploited, nonetheless it constitutes a compelling use case because it entails a physical system whose reactions to gestures are embodied. It is noteworthy, however, that we consider HRC as a motivating scenario, but our analysis and conclusions are by no means limited to it. The descriptions of the gestures in papers outlined in the Table have been elaborated by the authors of this paper on the basis of the papers themselves since usually gestures are not accurately described. This is a symptom of a deeper problem in this research field, i.e., an almost complete lack of standards, and therefore a concrete difficulty in building on top of the current state-of-the-art. This is even more acute in HCI scenarios.

Current literature lacks: (i) a common agreement of what a gesture is, which we aim at addressing in the context of HMI interfaces and not limited to HRI; (ii) a comprehensive taxonomy describing gestures in their relations with a generic intelligent system, and we provide a new gesture taxonomy based on a reasoned analysis of existing ones; (iii) a clear analysis of the involved problems in using gestures for HMI, which we better formalise as the *Gestural Interaction for User Interfaces* (GI-UI) problem. As a consequence the thesis is organised as follow. Chapter 2 aims at laying the ground with a definition of what a gesture is, introduces a multi-modal taxonomy of gestures, discussed the problem as entailed by gesture-based HMI and present a classification of the existing literature exploiting all the conceptual tool developed so far. Chapter 3 describes the usage of discrete gestures in HRI for industrial application and chapter 4 describes the usage of continuous gesture for

[3]This is an intuitive introduction to the concept of discrete and continuous gestures that will be formalised in the next Section.

programming by demonstration, a programming paradigm particularly useful for industrial robots[4]. Conclusion follows.

[4]The work carried on for this thesis resulted in different publications. In particular: the content of Chapter 2 has been submitted for review to IEEE Transaction of Cybernetics; the gesture recognition method introduced in Chapter 3 have been published in the IEEE International Conference on Robot & Human Communication [27]; and the content of Chapter 4 have been published in the International Journal of Social Robotics [28].

Chapter 2

Background & Problem Statement

In HMI, the *user interface* is a system, composed either by physical or software components, which allows someone to use a machine or an intelligent system. Obviously enough, the kinds of such machines or intelligent systems someone can interact with are countless, and vary from such physical systems as robots, to disembodied software applications. In interfaces which we can term as *classical*, user interaction is mediated by physical tools, e.g., a keyboard, and usually feedback is provided as a reasoned combination or sequence of visual stimuli, e.g., in the case of a graphical user interface (GUI).

In this thesis, we focus on gesture-based interaction for user interfaces, and in particular, we narrow down our attention on its technological requirements. In GUIs, a user directly interacts with the machine or intelligent system by means of gestures, although a physical device is still needed to perceive the gesture. As far as feedback is concerned, the classical approach could be seen as limited in different situations, e.g., while tele-operating a robot visual feedback may be directly provided by robot motion, and therefore a GUI may not be strictly necessary. Moreover, the concept of GUI is evolving because of the introduction of new visualisation techniques, for instance, those related to virtual or augmented reality, which as a matter of fact can be considered a whole separate field of research. For these reasons, and to devote much attention to a general analysis of gesture-based interaction, we decide to consider out of scope for this thesis how system feedback is conveyed to the user.

Interfaces mediated by tools such as keyboards or teach pendants, joysticks, and switches meant at controlling industrial robots are characterised by operations defined by the class the tool belongs to, their specific layout, and the user experience layer implementing the interface logic and a context-based feedback. Similarly, the operation of a gesture-based interface is defined by different albeit correlated components, either physical or disembodied, which define how a gesture is perceived, which gestures the system is expected to react to,

and how. The first step in the analysis of gesture-based interaction is obviously the definition of what a gesture is, how it can be characterised, and how such characterisation affects the structure of a gesture-based interface.

2.1 Gesture Definition

In general terms, it is not possible to provide an overall definition of what a gesture is, but it is possible to define it in the narrow scenario of HMI [29]. The notion of “gesture” results intuitive and, probably for this reason, in the literature the majority of works aiming at developing techniques and conceptual frameworks for gesture-based HMI do not explicitly define it. However, from an analysis of current state-of-the-art literature, we can extrapolate that, in HMI scenarios, gestures have been defined as *trajectories* [29, 30] of *body motion* [31–33] or *poses* [34] *performed intentionally* [35, 36] with the intent of *conveying meaningful information* or *interacting with the environment* [31, 34, 32, 33]. Therefore, we can summarise state-of-the-art definitions by giving ours:

> *Gestures are body actions that humans intentionally perform to affect the behaviour of an intelligent system.*

In providing this definition we have tried to be the more general as possible, with the aim of including all the different aspects which are present directly or indirectly in the literature[1]. The usage of "body actions", substituting motions or poses, is meant at yielding a general definition without focusing on a particular kind of gesture. Similarly, references to a particular body part, although in the literature gestures are usually defined for upper limbs, have been neglected in favour of a broader definition. Furthermore, the usage of the word "intentionally" is due to the context of our definition. In fact, functional HMI implies that users are aware that the system they are interacting with is monitoring their actions, and a gesture, to be meaningful for the interaction, should be performed intentionally. Therefore, this aspect should be included in the definition, since the machine should be able to distinguish between gestures and unintentional actions. The generic reference to the "behaviour" of an intelligent system allows us to consider a definition encompassing different applications and different mapping about how a gesture could affect the system.

[1] The focus of the definition is the action performed by a human and not the effects that it could have on other sensory media such as, for example, sound waves. Therefore vocal utterances are not gestures.

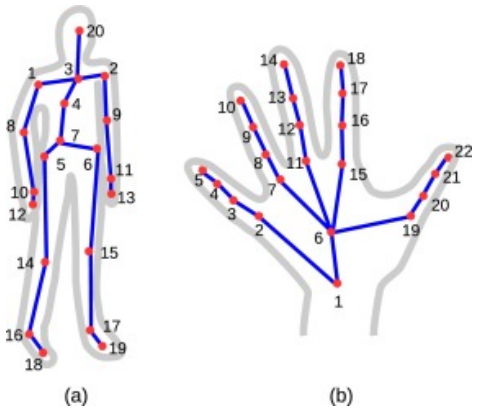


Figure 2.1 full-body (a) and hand (b) skeleton [37].

Inspired by those references whereby gestures are defined as trajectories in space, we extrapolated an analytical definition. If we represent the human body using a skeleton model such as the one in Figure 2.1, we can define human body postures using a joint status vector:

$$\boldsymbol{q}(t) = \{q_1, \dots q_i, \dots q_n\}$$

where q_i is the general joint angle between two consecutive skeleton links (lying on the plane where the two links reside), and n is the number of the considered joint angles in the skeleton. Furthermore, we can define $\dot{\boldsymbol{q}}(t)$ and $\ddot{\boldsymbol{q}}(t)$ as the angular velocities and accelerations of each joint, respectively. Then, we can define a trajectory in a n-dimensional space containing joint angles, their angular velocities and accelerations as:

$$\boldsymbol{\tau}(t_s, t_e) = \{\boldsymbol{q}(t_s) \dots \boldsymbol{q}(t_e), \dot{\boldsymbol{q}}(t_s) \dots \dot{\boldsymbol{q}}(t_e), \ddot{\boldsymbol{q}}(t_s) \dots \ddot{\boldsymbol{q}}(t_e)\}$$

where t_s is the time instant in which the trajectory starts and t_e is when it ends. Therefore:

> *Gestures are trajectories $\boldsymbol{\tau}(t_s, t_e)$ that human intentionally perform to affect the behaviour of an intelligent system.*

The substitution of the notion of "action" with that of a "trajectory" does not alter the generality of the definition as posed above. In fact, as we will see in the next Section, while trajectories represent a motion naturally, they can be used to describe poses as well. It is worth noting that our definition is articulated in two components, namely the trajectory, and the behaviour of the intelligent system. This conceptual separation will prove to be particularly useful in the next Section when we will define gesture taxonomies.

2.2 A Gesture Taxonomy

2.2.1 Analysis of Existing Taxonomies

Being able to establish appropriate gesture taxonomies is fundamental to properly define the kind of gestures a given HMI system is employing, and to have a clear understanding of the issues that such interface should address. Although in the literature this information is usually overlooked, characterising the kind of considered gestures is fundamental because it influences the synthesis of a well-defined problem statement for gesture-based interaction with machines. Before designing gesture-based interfaces, it is necessary to envisage a new taxonomy or to select an existing one. Gestures are complex elements of an interface and can be described from many perspectives and considering different characteristics. In this Section, we analyse five characterising features discussed in the literature to propose conceptualisations of gesture taxonomies, namely *time*, *context*, *level of instruction*, *body part*, and *spatial influence*.

The first gesture characteristic we consider is related to its temporal duration. From the temporal perspective, it is almost universally accepted that gestures can be classified as *static* or *dynamic* [38, 39, 31, 40–43, 11, 33]. Definitions can vary, but as a general rule-of-thumb static gestures are characterised by *poses*, while dynamic ones by *motions*. In the literature, dynamic gestures have been described as composed of three phases, namely *pre-stroke* (or preparation), *stroke*, and *post-stroke* (or retraction) [44, 31]. In the pre-stroke phase, a human moves to the starting pose associated with the gesture, during stroke the salient movement of the gesture happens, while in post-stroke either a resting position or the starting pose of the next gesture is reached. Two additional phases can be added [45], namely *pre-stroke hold* (if stroke is delayed), and *post-stroke hold* (if post-stroke is delayed).

The second gesture characteristic we discuss is related to the context associated with a gesture execution. Context-oriented taxonomies generated in semiotic studies have been adopted in HMI, and from time to time adapted to specific scenarios. However, the lack of a shared taxonomy leads to an heterogeneous and quite diversified literature, where the same concept, notion, or term can have a varied meaning. Nevertheless, a few similarities can be observed, which can contribute to a principled definition of a gesture. On a general basis, the dichotomy between gestures meant at communication (i.e., *communicative* gestures), and those targeted at manipulation (i.e., *manipulative* gestures) is widely accepted [44, 29, 46–48].

As an example, Quek *et al.* [44] provides a compelling example of the differences between them:

> *An orchestral conductor's hand motions are intended to communicate temporal, affective and interpretive information to the orchestra. A pianist's hand movements are meant to perturb the ivories. While it may be possible to observe the pianist's hands, the hand and finger movements are not meant to communicate with anyone.*

A recent, visual depiction on the communicative power of hand and arm gestures postulated by Quek *et al.* [44] as implemented in robots can be found in Alter 3, a humanoid torso from Ishiguro Labs characterised by human-like appearances acting as an orchestral conductor. The literature presents extensive classifications for communicative gestures, but only to a lesser extent for manipulative ones [44, 40, 49, 48].

Communicative gestures are further classified in [44] as *symbols* and *acts*. Symbols are gestures serving a linguistic role, and therefore requiring a common (typically mediated also by culture) background shared between the agent performing the gesture and the one having to interpret it, being the latter a human or a machine. In this context, a symbol can be *referential* or *modalising*:

- referential gestures are irrespective of other (active or inactive) communication channels and have a direct mapping into language significants or even words, e.g., rubbing the index and the thumb refers to money;
- modalizing gestures complement other active communication channels to add further information, e.g., holding hands apart while talking about an object may imply that it is long or short.

Acts are gestures whereby the associated motion is directly connected with the intended meaning. An act can be classified as being *mimetic* or *deictic*:

- mimetic gestures represent common sense or familiar concepts usually performing a pantomime, e.g., mimicking the lighting up of a cigarette to ask for a lighter;
- deictic gestures, also known as pointing gestures, can be further classified as *specific* (if used to select a given object), *generic* (if used to identify a class of objects by pointing to one of them), and *metonymic* (when pointing to an object to refer to some entity related to it or its function).

Stern *et al.* [40] extend the previous classification of manipulative and communicative gestures with other two kinds of gestures, namely *control* and *conversational* gestures:

- control gestures are used to command real or virtual objects, e.g., pointing gestures used to command a robot to pick up an object;
- conversational gestures occur during verbal interaction and refer to speech-related content.

Rojas-Muñoz and Wachs [49], introduce the so-called MAGIC gestural taxonomy, and divide gestures in *communicative*, *manipulative*, *Butterworth's*, and *regulatory*. While communicative and manipulative gestures have meanings similar to the descriptions given above, Butterworth's gestures are meant at signalling a failure in speech, e.g., the gesticulation of someone trying to recall or articulate a word, whereas regulatory gestures help control and better understand a conversation, e.g., understanding whose turn is to speak. It is noteworthy that the MAGIC taxonomy describes also nested classifications for communicative and regulator gestures. However, since no clear description is provided for subclasses, we do not detail the analysis here.

Karam and Schraefel [39] propose a slightly different taxonomy, dividing gestures in deictic, manipulative, *semaphoric*, *gesticulative*, *linguistic*, and with *multiple styles*:

- as also discussed above, deictic gestures are related to pointing movements used to establish the identity or the spatial position of an object;
- manipulative gestures are meant at controlling any object by mapping the gesture movement to the location and pose of the object;
- semaphoric gestures require a stylised dictionary of static or dynamic gestures usually associated with a command;
- gesticulation motions are hand movements performed while a person is speaking;
- linguistic gestures are meant at composing sentences in a language, e.g., in case of sign language for deaf people;
- multiple styles are gestures without a specific focus and are composed of a variety of other kinds of gestures.

Finally, Vuletic *et al.* [48] try to uniform taxonomies already present in the literature. Their proposed taxonomy recognises the difference between manipulative and communicative

gestures. However, they divide communicative gestures in *independent* and *speech-related* ones. Whilst independent gestures include symbolic, semaphoric, and pantomimic gestures, as described above, speech-related gestures complement speech contents, and are divided into *iconic* (further divided into *pictographic*, *spatiographic*, *kinematographic*, and *metaphoric*), modalising, *cohesive*, Butterworth's, *adaptive*, and deictic. Modalising, Butterworth's, and deictic gestures have been described above, whereas:

- iconic gestures complement information conveyed by speech to better illustrate what has been said, an example being a hand rolling motion while describing a rolling stone down a hill, and are further divided in *pictographic*, i.e., describing shapes, *spatiographic*, i.e., describing spatial relationships, *kinematographic*, i.e., describing actions, and *metaphoric*, i.e., for such abstract concepts as a cutting-like gesture used to interrupt a conversation;
- cohesive gestures are performed to refer to concepts previously introduced in the conversation;
- adaptive gestures are involuntary motions performed to release body tension.

The third gesture characteristic we focus on, i.e., the level of instruction, has been introduced in the review by Vuletic *et al.* [48]. From the perspective put forth in that review, gesture-based HMI can be classified in *prescribes*, i.e., those interfaces in which a gesture dictionary is required, and *free-form*, i.e., a dictionary is not necessary. It is noteworthy that here a focus shift is present. In fact, all previous taxonomies are gesture-centred, while when considering the level of instruction and the taxonomy proposed in the review, what is classified is the whole interaction.

As a side result, the work in [48] also highlights the difficulty faced by the scientific community to build on top of existing work because of a lack of datasets, and a clear problem statement in gesture-based HMI research. Although it provides an excellent contribution in reviewing gesture-based HMI approaches, nonetheless it does not provide any new common framework to be shared by the whole community and therefore adopted for future research.

In our opinion, the features characterising gestures in HMI interfaces influence the problem statement, but not all differences in gesture features imply a difference in the problem statement itself. Just for the sake of the argument, a machine used to interpret sign languages or semaphoric gestures should face the same engineering issues, namely human actions be sensed continuously, relevant part of the data stream be isolated to prevent unintentional movements to be recognised as actions, the selected data segment be classified

according to a dictionary, and a meaning be associated with the performed action, whereas for a semaphoric gesture each class would have to be associated with a given command. On the contrary, if we consider a manipulative gesture, human actions should be continuously sensed, the data stream be processed to extract relevant features, e.g., the hand orientation, and those features be continuously mapped to a relevant machine function, e.g., the orientation of an object in a 3D virtual environment. These are two cases whereby the selected gestures affect data processing, but other differences can affect other aspects of the problem statement. For example, a gesture performed with an arm can be sensed using a wrist-worn Inertial Measurement Unit (IMU) but in order to sense a gesture performed with the fingers, another device should be used.

The latter example allows us to introduce the fourth gesture characteristic we consider, i.e., the body part used to perform a gesture. This characteristic is usually ignored in the literature since gestures, especially in HMI interfaces, are implicitly considered to be performed with upper limbs. For this reason, apart from very few cases, in the literature, the specification of the used body part is completely neglected, which can be a major cause of confusion. Although it is implicit that gestures are performed with upper limbs, it is not always clear if a specific gesture requires using an arm, an hand or the fingers. Only a few works clearly specify their focus on hand-based gestures [50] or organise gestures as *hand-arm*, *head-face* and *body* [31].

Finally, the last gesture characteristic described and used to classify gesture-based interaction is spatial influence [31]. According to this perspective, the same gesture may have a different meaning depending on where it is performed, location framing a sort of contextual information. For example, pressing a mid-air virtual button it is always performed in the same way, however, the position of the hand determines which button is actually pressed [51].

2.2.2 A reasoned Taxonomy of Gestures

In this Section, building on the knowledge about available, state-of-the-art, gesture taxonomies introduced above, we can define and describe the components of the taxonomy we introduce in this work. As a preliminary note, we stress out that our taxonomy is focused on the issues that each gesture characterisation implies, where no sociological connotation is involved. In the discussion that follows, we first define the main characteristics, and then we analyse how they affect the problem statement. We consider four features, which we refer to as *effect*, *time*, *focus*, and *space*.

The **effect** is aimed at describing how a gesture is going to affect the machine the human is interacting with. If we refer to the function mapping a gesture $\boldsymbol{\tau}$ to the system state as $f(\boldsymbol{\tau})$, its effect describes the mapping itself. According to their effect, gestures can be either continuous or discrete. The information yield by a continuous gesture is mapped at each time instant to a change in the interface state, while for discrete gestures such change is *atomically* associated with the whole gesture. A gesture effect has also implications on the system state, which should be continuous or discrete according to the selected kind of gestures. Let us define $S_c \in \mathbb{R}^n$ as an n-dimensional continuous state, and $S_d \in \mathbb{S}$ as a discrete state, where $\mathbb{S}$ the set of all possible, countable, discrete states. For continuous gestures, $f(\boldsymbol{\tau})$ has domain $\boldsymbol{\tau}^{(t_e - t_s)}$ and codomain $S_c^{(t_e - t_s)}$, whereas for discrete gestures the codomain is simply the discrete state S_d. The two time instances t_s and t_e, defined in Section 2.1, refer to the start and end events of a gesture. Obviously, and adopting a pragmatic perspective, S_c can not be ideally *continuous*, since its variations are related to the frequency at which the gesture is sampled by the employed sensors. If we consider a parallelism with the mouse-keyboard interface, continuous gestures are mouse movements while discrete gestures are the keystrokes. It is noteworthy that although the classification between continuous and discrete gestures could recall the dichotomy between manipulative and communicative gestures, they are intrinsically different. The context characteristic described above has been used to describe the intended meaning of gestures, for example, if they are aimed at manipulation or communication, while here we are describing the nature of the gesture effect. As an example, let us consider two different communicative gestures, i.e., semaphoric and language gestures. The former class includes gestures whose effect is defined on the basis of a synthetic dictionary associated with the execution of a command, while the latter is related to gestures characterised by a linguistic role. This difference in their usage characterises their distinction from a contextual perspective. However, from an effect-related perspective, as long as these gestures are associated with a discrete command and a word, respectively, they are both discrete gestures.

In continuity with state-of-the-art taxonomies, the **time** characteristic classifies gestures in static or dynamic. However, differently from what is typically postulated in the literature, where phases are only foreseen for dynamic gestures, we organise all gestures in three phases, i.e., *preparation*, *stroke* and *retraction*. A classification of a gesture as static or dynamic is aimed at describing the human behaviour during the stroke phase. Therefore, in our case, a person performing a static gesture first moves to the desired pose (preparation), then keeps a static pose for an arbitrary amount of time (stroke), and finally returns to the rest position

(retraction). If we refer to the gesture definition introduced in Section 2.1, we can specify that for dynamic gestures it exists at least one $q_i \in \boldsymbol{q}$ such that $\dot{q}_i \neq 0$, whereas for static gestures it holds that $\dot{\boldsymbol{q}}(t_s) = \ddot{\boldsymbol{q}}(t_s) = 0$. Please notice that t_s and t_e refer here to the starting and ending time of the stroke phase.

Our taxonomy enforces that static gestures cannot be considered continuous. In fact, although a static gesture *per se* may be considered continuous, it is not possible to associate a static gesture to a continuous system state. To make this possible, it would be necessary to define an infinite amount of postures for each possible instance element of the continuous system state, and this is not only impractical but *de facto* impossible.

In the taxonomy, **focus** describes which body parts are relevant for a gesture. The name of this characteristic has been chosen to highlight the fact that the relevance of a body part is determined *a priori* by the HMI requirements, and it does not depend on the specific action. For example, in an application whereby the focus is on the arm, whichever gesture is performed by the hand is ignored. The focus characteristic does not divide gestures in distinct classes, but describes each gesture using the names of the relevant body parts. Referring again to our gesture definition, the focus on a specific body part implies that we are interested only in the status of the joints relevant to that body part, e.g., in an hand gesture, the interested joints are the ones modelling the wrist, essentially. For this reason, a waving gesture is first of all an arm gesture, since the motion is generated by arm joints, and can be considered a combined arm-hand gesture if we are interested even in the status of the wrist.

Considering only relevant body parts helps reduce the size of an associated classification problem. In fact, if just a subset $\boldsymbol{p}(t) \subseteq \boldsymbol{q}(t)$ is needed to represent a gesture, then the trajectory associated with that gesture can be redefined as $\boldsymbol{\tau}_p(t_s, t_e)$. It is noteworthy that focus can also model gestures referring to multiple, or even non directly connected, body parts, e.g., bi-manual gestures [52].

The last gesture characteristic we include, namely **space**, likewise other taxonomies described in the previous Section, determines whether the meaning associated with a gesture is influenced by the physical location of the body part performing it. According to this distinction, gestures can be spatially related or unrelated. We have previously seen how the discrete gesture of a mid-air virtual button pressure can be spatially related. Similarly, a manipulative gesture for dragging and dropping a virtual object is spatially related, however a manipulative gesture for controlling the velocity of a mobile robot by means of the arm inclination and rotation, as done in [20], is spatially unrelated.

Table 2.1 Summary of the influence that gestures can have on sensing, data processing, and system reaction according to the gesture type.

Characteristic	*Sensing*	*Data Processing*	*System Reaction*
Effect	None	Major	Major
Time	Major	Minor	None
Focus	Minor	Minor	None
Space	Major	Minor	None

The four characteristics described in the Section make up our gesture taxonomy. A human action can be composed of multiple gestures. Getting back to the hand waving example, if we want to describe the fact that the fingers are spread while the arm is moving, then the gesture can be classified as a *discrete gesture dynamic-arm and static-fingers*, otherwise it could be simply a *discrete gesture dynamic-arm*.

2.3 Problem statement

In the previous Section, we have introduced our notion of gesture and described its characterisation in the form a taxonomy. Here, we aim at structuring the problem of gesture-based interfaces for human-machine interaction. As anticipated in the Introduction, we refer to this problem as Gestural Interaction for User Interfaces, and we refer to it as GI-UI. It is noteworthy that the problem statement we put forth in this thesis has an operational and engineering nature. As such, it is structured as the interplay among three, interrelated, sub-problems, i.e., *sensing*, *data processing* and *system reaction*. Although the structure of the problem statement is fixed and well-defined, i.e., sensing influences data processing, which is mapped to a certain system reaction, the design choices of sub-problems are tightly related to the gestures an HMI interface can consider. In the following paragraphs, we explore the relationships among these sub-problems more in-depth.

2.3.1 Sensing

The kind of sensors used to collect data about gestures depend on the particular application requirements, e.g., privacy [53], price [54] or computational efficiency [55]. Two conflicting paradigms affect gesture sensing, namely *come as you are* and *wearability* [54]. The phrasing "come as you are" refers to a system design whereby its users should not *wear* anything to interact with the machine. On the contrary, wearability refers to systems assuming that users are wearing such interaction tools as data gloves or smartwatches, i.e., they bring the

HMI interface with them. This distinction is typically used to classify sensing approaches as non-image-based and image-based [56]. Non-image-based methods include both wearable devices, such as instrumented gloves, wristbands and body suits, and non-wearable devices, using radio frequency or electric field sensing. Image-based approaches are probably one of the most widely explored research fields, as many literature reviews address specific issues and solutions related to this sensing modality only [41, 57]. However, the list of sensing solutions [48] is wide and not easy to analyse nor classify. Our aim is not to list different approaches or to propose a classification. Instead, what we want to highlight is the influence that the kind of gestures we consider can have in the choice of the sensing device, while taking into consideration that – as we have seen previously – other factors may affect our choice as well. A visual representation of the extent to which gesture characteristics may influence the sensing strategy is presented in Table 2.1. According to our observations, the temporal and spatial characteristics have a significant influence on the sensing strategy. In fact, static and dynamic gestures can generally be sensed using the same sensors. However, some sensors are more suited for specific kinds of gestures. For instance, in stationary conditions, an accelerometer can be used to determine its inclination with respect to the horizontal plane, therefore making it easy to monitor simple static gestures [58]. Instead, if we consider a dynamic gesture the usage of accelerometers is not enough to track the sensor pose and information from other sensors, such as gyroscopes or magnetometers, should be integrated [59]. Similarly, in order to recognise a spatial gesture, such as a pointing gesture, one must select a sensor allowing for the extraction of spatial information, e.g., a camera [19]. Finally, we can observe that the influence of focus on the sensing strategy is limited, and it is mainly associated with wearable devices. In fact, depending on the gesture focus, the sensor should be placed to have visibility of the movement, e.g., for a gesture involving fingers IMUs should be placed on the fingers and not on the arm [60].

2.3.2 Data processing

Many factors can influence data processing. One in particular, however, drastically changes the nature of the problem an HMI designer must solve, i.e., the effect, as presented in Table 2.1. As a matter of fact, depending whether we consider continuous or discrete gestures, the problem statement completely changes. In the case of continuous gestures, at each instant its representation must be directly associated with a machine reaction, or function. In certain cases, this is possible using raw data [20]. However, data should be processed to extract relevant, possibly *semantic* features, such as the 2D position of the hand with respect to an

image plane [21]. As it is customary, we refer to this procedure as *feature extraction*. Instead, for discrete gestures feature extraction is part of a more complex problem. Raw data or the extracted features should be analysed to determine the start and the end points of the gesture, and to classify it according to a predefined dictionary [61], a process usually termed *gesture recognition*.

Gesture recognition has been described as the process whereby specific gestures are recognised and interpreted [31, 36]. Some studies consider gesture recognition as a three-step process, composed of identification, tracking and classification [56], or alternatively of detection, feature extraction and classification [34]. Other studies consider it a two-step process, made up of detection and classification [62]. In the literature, the characterisation of the gesture recognition problem is highly influenced by the considered sensors and the target application. Here, we try to give a general definition. The gesture recognition problem is the process that, given sensory data and a dictionary of discrete gestures, determines whether any gesture has occurred, and which one. To this aim, sensory data are supposed to undergo three computational steps, namely pre-processing and feature extraction, detection and classification. In the first phase, raw sensory data are manipulated to de-noise and to extract relevant features. In the detection phase (also referred to as segmentation or spotting), filtered data are analysed to determine the occurrence of a gesture, and its start and end moments. Detection is usually performed using filtering techniques or threshold-based mechanisms [56]. The classification phase determines which gesture present in the dictionary has been performed. Often, classification is probabilistic, and together with the label it returns a confidence value [61]. The literature has explored different approaches for gesture recognition, encompassing purely model-based techniques [12] to machine learning [22], whereby the adopted techniques are highly dependent on the data source. The order used to discuss the three phases is not binding, especially with reference to detection and classification. In fact, gesture detection can be direct, when it is performed on sensory data, or indirect, when it is performed on classification results [63].

As shown in Table 2.1, other gesture characteristics can have minor effects on data processing. Static and dynamic gestures imply intrinsically different data, since gestures belonging to the former class are not related to time, whereas dynamic gestures are. Therefore, the techniques adopted to process static and dynamic gestures are different. Similarly, the focus of a gesture influences the kind of collected data, or the features that the system should extract, i.e., *head* or hand gestures extracted from video streams imply a different data processing pipeline. Finally, spatially related gestures require the feature extraction process to consider the position of the body part performing the gesture as a feature.

2.3.3 System Reaction

The way the HMI system responds and acts to a given gesture varies depending on the application. As an example, it may consist of a switch in an interface menu (HCI) [64], or a velocity command for a mobile robot (HRI) [20]. Obviously enough, system responses are the results of a mapping between data processing and machine behaviours. This mapping serves as a sort of semantic attribution to gestures in the context of the interaction process. As it can be seen in Table 2.1, the reaction is to a large extent agnostic to gesture characteristics, since their effects are absorbed by the sensing and the data processing phases. The effect characteristic affects of course system reaction, since continuous and discrete gestures by their own nature implies continuous and discrete system functions, respectively.

An obvious topic to deal with when considering machine behaviours in response to human gestures in HMI interfaces is how the machine can expose a natural and intuitive interface to its users. Many studies have been carried out in the past decades, and the recent urge to design and develop technologies for the consumer market has further increased this trend [48]. Therefore, we decided not to consider this topic in the survey, and we deem it out of scope. The interested reader is referred to [54]. However, we focus in the next Section on those interaction-related traits that can be directly linked to the taxonomy we introduce in Section 2.2, and to the problem statement presented in 2.3.

2.4 The human factor

2.4.1 Requirements

In the previous Sections, we have considered humans as mere entities performing gestures, and we have discussed HMI processes only from a machine perspective. However, human presence is fundamental and can influence the design choices taken while developing solutions for the GI-UI problem as a whole, although in particular system reaction is affected. Many requirements dealing with human factors, and well-aligned with the GI-UI problem, have been identified in the literature [54]. These include *responsiveness*, *user adaptability and feedback*, *learnability*, *accuracy*, *low mental load*, *intuitiveness*, *comfort* and *lexicon size*. All these requirements should be taken into account while designing an HMI process. Although they do not modify the general structure of the problem, they can surely enforce certain solutions with respect to others.

Responsiveness is a metric typically associated with the dynamic flow of the HMI interface. As a rule-of-thumb, the *response time*, i.e., the time interval between user input

and system reaction, should be as low as possible. The overall system should be carefully designed to react to gestural stimuli as fast as possible [38, 65]. Even a non expert user can easily perceive if the reaction time increases, and if it exceeds a psychological and cognitive threshold, user experience and satisfaction can be seriously disrupted [66]. Different applications may imply different such thresholds. For general-purpose applications, classical design methodologies in HMI recommend response times lower than 100 ms. However, recent studies found out that the acceptable latency may be even lower, especially for interactions resembling physical ones [66]. This requirement may prevent designers to adopt sensors generating lots of data, or particularly heavy data, especially when the processing hardware is not top performance, such as in edge or field applications or when the associated processing techniques are characterised by a high level of complexity.

Especially in HMI interfaces employing discrete gestures, the system is supposed to distinguish among a limited number of gestures. Therefore, depending on the application, the interface should be capable of *adapting* to certain user specific traits, either cognitive or physical [67]. Many contributions to the literature highlight the need for gesture-based interfaces to allow a user to personalise the (set of) gestures used in the interaction [50]. This requirement has an obvious technological consequence, i.e., the techniques used to model gestures, as well as those related to its run-time processing, must allow for an easy-to-attain adaptation even for non expert users. Most likely, this would involve the possibility for the system to learn from experience [13].

Furthermore, the HMI interface is expected not only to react to the detected gesture minimising the response time, but also to provide the user with an adequate *feedback* about the correctness of the gesture itself. Lack of direct feedback in HMI can ultimately lead to a lack of trust in the system and a deranged user experience [54]. Among the methods currently exploited to provide feedback to users, we can mention classical GUIs [21], haptics [68], augmented [69] and virtual [70] reality.

For a real-world use, a gesture-based interface system correctly interpret gestures in input with *accuracy* levels close to 100% [50]. This is not only preferred from an engineering perspective, it is also of the utmost importance for an interface aimed at being used in everyday conditions. As a direct, although not obvious consequence of this requirement, gestures should be designed and modelled with two somewhat contrasting objectives in mind. On the one hand, they should be such to maximise the ease of classification, i.e., enforcing accuracy; on the other hand, they should preserve their intuitiveness [22]. In accordance with our definition of continuous and discrete gestures, the notion of accuracy for these two kinds of gestures is different. In fact, while for continuous gestures it is important to properly

estimate the relevant features, and therefore depending by the specific feature the error metric can change, in the case of discrete gestures what matters most is the recognition rate, usually expressed using such parameters as accuracy, precision, recall, and F1 score [71].

The requirements associated with learnability, mental load, comfort and intuitiveness are strictly intertwined. The set of gestures should be easy to learn, whereas the training time for new users should be brief [72]. While in a general sense the HMI process, and the interface, in particular, should not heavily impact on the user mental load [54], it is widely accepted that gesture-based HMI can reduce it to a great extent [47]. As a consequence, the selected gestures should be simple and brief, which is expected to enforce also learnability. Complex gestures, or gestures that imply intense muscle activity should be avoided to guarantee user comfort, especially if the gesture-based interface is designed to be used for a prolonged time [54] Finally, selected gestures should have an intuitive association with the expected system behaviour. This, in turn, facilitates learnability and reduces the user mental load [73].

The capability of a gesture-based HMI interface to interpret a high number of gestures, i.e., the lexicon size, can be of the utmost importance for its usability and effectiveness, an obvious example being the automated interpretation of the sign language [74]. However, increasing the lexicon size is in contrast with the learnability and low mental load requirements, because a bigger dictionary is more difficult to learn and recall [75]. Furthermore, a bigger dictionary may imply lower performance in data processing, because of increased problem complexity. As a consequence, the size of the dictionary should be mediated by designers considering all these factors.

In summary, we can observe that many of the considered requirements refer to the selected gestures and to the implications that their choice can have both on user experience and the problem solution. We can conclude from this brief overview that the gesture choice and the associated system behaviour are fundamental. The lexicon size is specifically related to the dictionary composition. As we have previously mentioned, dictionary design is a problem associated with gesture recognition and therefore to the specific case of discrete gestures. In the next Section we focus on what a gesture dictionary is, and how we can design it.

Dictionary

In general terms, a dictionary (or vocabulary) is the set of words making up a language and the associated meaning. In our case, the dictionary contains the set of gestures that the interface is able to interpret, i.e., for each gesture a description of the trajectory $\boldsymbol{\tau}$, as well as the associated elicited behaviour. The definition of a gesture dictionary is an issue

related only to semaphoric gestures, as introduced in Section 2.2. For all other gestures a dictionary is still necessary, but as a matter of fact it is given by the context. For example, the development of an interface for the translation of sign language does not require the definition of a new dictionary since it is already available.

For semaphoric gestures, the dictionary can be defined as a set of matched pairs of commands and their gestural expression [40]. As we already discussed in the previous Section, a dictionary should made up of gestures that are intuitive, physically easy to perform, easy to be recognised, easy to learn, and easy to remember [40]. In order to build dictionaries, for a given application and to meet certain requirements, many alternatives have been explored in the literature. Different approaches can be used to build dictionaries. The first approach implies using questionnaires whereby volunteers are asked to sketch the gestures they consider most appropriate for the specific application. It has been observed that if the gestures are intuitive enough, a user can easily adapt to gestures defined by others [50]. According to the second approach, volunteers can be required to mimic a gesture-based interaction for a specific application, and as a consequence the dictionary is built based on the experimenter observations [76]. The third approach involves the usage of Wizard of Oz experiments [77]. In this case, volunteers are tasked with solving an interaction problem with a machine they suppose to be autonomous. Instead, the machine is manually operated by the experimenter to mimic its behaviour in response to human gestures as if the machine was autonomous. Then, the observations of gestures done by volunteers can be used to define the dictionary.

Once the dictionary is designed, it is important to have tools to determine whether it satisfies the requirements listed above. This can be achieved by performing experiments and determining metrics to evaluate the considered characteristics [40]. Alternatively, a few studies have proposed the Vocabulary Acceptability Criteria (VAC) [73], which allows experts to evaluate gestures on the basis of six attributes, namely *iconicity*, i.e., how much a gesture recall the associated command, *simplicity*, *efficiency*, *compactness*, i.e., how much the gesture covers the space around the body, *salience*, i.e., how discriminating is a movement , and *economy*, i.e., related to the movement magnitude [73]. Although it is nowadays clear what are the requirements that should characterise a dictionary, no standard exists yet for dictionary design and evaluation.

Another problem related to gesture dictionaries is how to effectively describe a gesture. This is important for reproducible research. Often, original data sets are not available, and the only way to reproduce work done by others is to collect a new data set following the same experimental procedure. However, it is not always clear how gestures have been performed.

Gesture taxonomies can help disambiguate in some scenarios, but they may not be sufficient. Drawings in the literature are often used to describe the gestures, and are preferred to videos because they can be easily shared and printed. We believe that for now the drawing option, combined with a text-based description, is the most convenient solution. However, this still remains an open issue.

2.5 Literature classification

As we observed above, the literature about gesture-based interaction is vast and heterogeneous. In the Section, we exploit the conceptual tools developed in the previous Sections to analyse and classify it. Table 2.2 includes all the reviewed articles classified using the taxonomy introduced in Section 2.2. Articles are ordered chronologically. All the columns of the Table are grouped into five categories:

- *Article info* includes two sub-columns, namely the article reference and the publication year.
- *Sensing* focuses on the employed sensing modalities, and reports multiple sensors whenever they are used.
- *Reaction* describes at a high level whether a certain work refers to HCI or HRI studies, i.e., whether the reaction involves a virtual or a physical system. Here we consider only those papers whereby such reaction is clearly described, or a validation is presented. Articles not fulfilling these requirements are classified as not having a system reaction.
- *Processing* includes two sub-columns, the former with the specification of the problem actually solved, i.e., recognition, feature extraction or classification, the latter with information about user-defined gestures.
- *Gestures* is the proper gestures classification, including the size of the dictionary for discrete gestures, as well as the effect, time, focus and spatial characteristics.

In the Table, the articles which consider distinctly discrete and continuous gestures are described using two rows for the gesture and processing groups. This happens even when an article elaborates on multiple dictionaries. If different gestures have different application scenarios, an extra row is added to the reaction column as well. The focus column can refer to more than one body part, on the basis of the specific article contents. The main factors we considered to determine the focus of the gestures related to a specific work

are the description of gestures, pictures or videos whereby gestures are shown, employed sensors and computational approaches. In the Table, when the focus-related column contains multiple entries, while the columns related to effect and time contain one entry only, the latter characterise the gesture of each of the referred body parts, an example being the row in [75] for a discrete, dynamic, finger/hand gesture. If the time column contains both the static and dynamic keywords, and just one body part is specified in the focus column, then the dictionary contains both static and dynamic gestures performed with that body part, e.g. in [78] one discrete, static, hand gesture and four discrete, dynamic, hand gestures are used. If both the focus and time columns contain two entries, this means that the dictionary is composed of gestures whereby one body part is static and the other is dynamic. The order is preserved in-between columns, e.g., in [79] are used discrete, static, fingers as well as dynamic hand gestures. This, of course, can be extended even to the case whereby effect, time and focus columns contain two keywords, e.g., in [80] discrete, static, fingers and continuous, dynamic, arm gestures. The *bi* tag has been added to the focus column in all cases addressing the processing problem simultaneously for different body parts. Works where bi-manual gestures are considered as the combination of two single limb gestures combined after the processing phase, as done for instance in [22], are not identified by this tag.

While performing the review of relevant literature, we have encountered a huge variety of meanings associated with the gesture recognition problem. Often, the term "gesture recognition" is used to refer to whichever procedure related to gesture-based sensing and processing. Nevertheless, and to the best of our knowledge, we did not encounter works that we could not represent using our taxonomy. Whilst the intent of this review is to include as many references as possible, a few articles have been discarded because of their lack of clarity in presentation. However, the idea of the authors is to keep this list updated on a dedicated website[2] accepting suggestions by the community.

More extensive and methodical reviews exist, which may be useful to perform relevant statistical analyses on the kind of adopted sensors, gesture types, and dictionary size. Therefore, we do not want to perform this kind of observations here. However, it is noteworthy to point out one single observation related to our framework. According to our problem statement, in order for the interface to react to a gesture-related stimulus, the processing phase should be completed. As a consequence, the problems to be solved are feature extraction for continuous gestures, and gesture recognition for discrete gestures. Consistently, in Table 2.2 each work that exhibits a reaction solves either feature extraction or gesture recognition.

[2]Web: https://acarfi.github.io/GesturalInteractionSurvey

Table 2.2 Classification of literature related to gesture-based interfaces.

Article info		*Sensing*			*Reaction*	*Processing*				*Gestures*		
Ref	*Year*	*Sensor (1)*	*Sensor (2)*	*Sensor (3)*		*Problem*	*User defined*	*Size*	*Effect*	*Time*	*Focus*	*Space*
[12]	1996	Stereo Monochrome Camera	-	-	HRI	Gesture Recognition	No	6	Discrete	Static	Arm	Yes
[81]	1996	Stereo Monochrome Camera	-	-	HRI	Feature Extraction	No	-	Continuous	Dynamic	Fingers	Yes
[80]	2000	Monochrome Camera	Infrared Illumination	Buttom	-	Gesture Recognition Gesture Recognition	No Yes	8 6	Discrete Continuous Discrete	Static Dynamic Dynamic	Fingers Arm (Bi) Fingers Hand	No No
[75]	2000	Camera	-	-	-	Gesture Recognition	No	5	Discrete	Dynamic	Finger Hand	No
[13]	2000	Camera	-	-	HRI	Gesture Recognition	Yes	-	Discrete	Dynamic	(Bi) Hand	No
[82]	2000	Electro-mechanical Strain Gauges	-	-	-	Gesture Classification	No	90	Discrete	Dynamic	(Bi) Fingers	No
[26]	2002	Camera	-	-	-	Gesture Classification	No	6	Discrete	Static	Fingers Hand	No
[83]	2003	Camera	-	-	-	Gesture Recognition	No	5	Discrete	Dynamic	Fingers Hand	No
[38]	2003	RGB Camera	-	-	-	Gesture Classification	No	6	Discrete	Static	Fingers Hand	No
[14]	2004	RGB Camera	-	-	HRI	Gesture Recognition Gesture Recognition	No No	8 2	Discrete Discrete	Static Dynamic	(Bi) Fingers Hand Head	No No
[72]	2004	Accelerometer	Button	-	-	Gesture Recognition	No	8	Discrete	Dynamic	Hand	No
[35]	2004	RGB Camera	-	-	HCI	Gesture Recognition	No	10	Discrete	Dynamic	(Bi) Arm Torso	No
[50]	2006	Accelerometer	Button	-	HCI	Gesture Recognition Feature Extraction	No No	8 -	Discrete Continuous	Dynamic Dynamic	Hand Hand	No No
[84]	2007	Stereo Camera	-	-	-	Gesture Recognition	No	1	Discrete	Dynamic	Hand	Yes
[78]	2007	Proximity Sensors	-	-	-	Gesture Recognition	No	5	Discrete	Dynamic Static	Hand	No
[85]	2007	Accelerometer	Button	-	-	Gesture Recognition	No	8	Discrete	Dynamic	Hand	No
[86]	2008	Accelerometer	Button	-	-	Gesture Recognition	No	5	Discrete	Dynamic	Hand	No
[15]	2009	Accelerometer	-	-	HRI	Gesture Classification Gesture Recognition	No No	12 2	Discrete Discrete	Dynamic Static	Arm Arm	No No
[87]	2009	Accelerometer	Button	-	HCI	Gesture Recognition Gesture Recognition	No Yes	8 -	Discrete Discrete	Dynamic Dynamic	Hand Hand	No No
[16]	2010	Accelerometer	-	-	HRI	Gesture Recognition	No	6	Discrete	Dynamic	Arm	No
[88]	2010	Accelerometer	Button	-	-	Gesture Recognition	No	18	Discrete	Dynamic	Hand	No
[89]	2011	MoCap	Button	-	HCI	Feature Extraction	No	-	Continuous	Dynamic	Arm	No
[90]	2011	Accelerometer	EMG	-	HCI	Gesture Recognition Gesture Recognition	No No	72 18	Discrete Discrete	Dynamic Dynamic Static	Hand Fingers Hand Fingers	No No
[91]	2012	Accelerometer	Button	Marker Based MoCap	-	Gesture Recognition	No	20	Discrete	Dynamic	Hand	No
[92]	2012	Accelerometer	-	-	-	Gesture Classification	No	8	Discrete	Dynamic	Hand	No
[64]	2012	Microphone	Speaker	-	HCI	Gesture Recognition	No	5	Discrete	Dynamic	(Bi) Hand	No
[93]	2012	RGB-D Camera	-	-	HCI	Gesture Recognition Feature Extraction	No No	4 -	Discrete Continuous	Dynamic Dynamic	Hand Hand	No Yes
[94]	2013	Accelerometer	Touch Screen	-	HCI	Gesture Recognition	No	8	Discrete	Dynamic	Arm	No
[95]	2013	Accelerometer	-	-	-	Gesture Classification	No	20	Discrete	Dynamic	Hand	No
[96]	2013	RGB-D Camera	-	-	HCI	Gesture Recognition	No	3	Continuous Discrete	Dynamic Static	Hand Fingers	Yes
[97]	2014	Accelerometer	Camera	-	-	Gesture Classification	No	10	Discrete	Dynamic	Hand	No
[17]	2014	RGB-D Camera	-	-	- - HRI	Gesture Recognition Gesture Recognition Gesture Recognition	No No No	5 12 3	Discrete Discrete Discrete	Dynamic Dynamic Dynamic	Arm Full Body Arm	No No No
[98]	2014	EMG	Accelerometer	-	HCI	Gesture Recognition Gesture Recognition	No No	4 15	Discrete Discrete	Static Static Dynamic	Fingers Hand Fingers Arm	No No
[65]	2014	Stereo Dynamic Vision Sensor	-	-	-	Gesture Recognition	No	11	Discrete	Dynamic	Hand	No
[99]	2014	RGB-D Camera	6-Axis IMU	-	-	Gesture Classification	No	5	Discrete	Dynamic Static	Arm	No
[100]	2014	Accelerometer	Button	-	-	Gesture Recognition	No	8	Discrete	Dynamic	Hand	No
[101]	2014	RGB-D Camera	-	-	-	Gesture Recognition	No	19	Discrete	Dynamic	Fingers Hand	No
[102]	2014	6-Axis IMU	-	-	-	Gesture Classification	No	9	Discrete	Dynamic	Hand	No
[103]	2014	Infrared Camera	-	-	HRI	Gesture Recognition	No	2	Continuous Discrete	Dynamic Static	Hand Fingers	No
[60]	2015	Accelerometer	-	-	-	Gesture Recognition	No	12	Discrete	Dynamic	Fingers	No
[104]	2015	Infrared Camera	-	-	HCI	Gesture Recognition	No	3	Discrete	Dynamic	Hand	No
[105]	2015	Accelerometer	-	-	-	Gesture Classification	No	7	Discrete	Dynamic	Hand	No
[18]	2015	RGB-D Camera	-	-	HRI	Gesture Recognition	No	10	Discrete	Dynamic	Arm	No

Table 2.2 Classification of literature related to gesture-based interfaces - continued.

Article		*Sensing*			*Reaction*	*Processing*				*Gestures*		
Ref	*Year*	*Sensor (1)*	*Sensor (2)*	*Sensor (3)*		*Problem*	*User defined*	*Size*	*Effect*	*Time*	*Focus*	*Space*
[30]	2015	Magnetic 3D Position Tracker	-	-	HCI	Gesture Recognition	No	11	Discrete	Dynamic	Hand	No
[106]	2015	9-Axis IMU	-	-	-	Gesture Recognition	No	10	Discrete	Dynamic	Hand	No
						Gesture Recognition	No	26	Discrete	Dynamic	Hand	No
						Gesture Recognition	No	8	Discrete	Dynamic	Hand	No
[107]	2015	EMG	6-Axis IMU	Button	-	Gesture Recognition	No	12	Discrete	Dynamic	Fingers Hand	No
[108]	2015	RGB-D Camera	-	-	-	Gesture Classification	No	10	Discrete	Static	Fingers	No
[109]	2015	RGB-D Camera Infrared Camera	-	-	HCI	Feature Extraction	No	-	Continuous	Dynamic	Hand Fingers	Yes
[110]	2015	9-axis IMU	-	-	-	Gesture Recognition	No	11	Discrete	Dynamic	Hand	No
[111]	2015	Infrared Camera	-	-	-	Feature Extraction	No	1	Discrete	Dynamic	Fingers	Yes
[74]	2016	7 Accelerometers	-	-	-	Gesture Classification	No	40	Discrete	Dynamic	Fingers Hand Arm	No
[112]	2016	Accelerometer	-	-	-	Gesture Recognition	No	5	Discrete	Dynamic	Arm	No
						Gesture Recognition	No	5	Discrete	Dynamic	Arm	No
[113]	2016	Accelerometer	Button	-	-	Gesture Recognition	No	9	Discrete	Dynamic	Hand	No
						Gesture Recognition	No	7	Discrete	Dynamic	Hand	No
[114]	2016	6-Axis IMU	-	-	-	Gesture Recognition	No	5	Discrete	Dynamic	Fingers	No
[115]	2016	Accelerometer	Button	-	-	Gesture Recognition	No	8	Discrete	Dynamic	Hand	No
						Gesture Recognition	No	16	Discrete	Dynamic	Hand	No
[116]	2016	6-Axis IMU	-	-	HCI	Gesture Recognition	No	6	Discrete	Dynamic	Hand	No
[117]	2016	RGB-D Camera	Stereo Infrared Camera	-	-	Gesture Classification	No	25	Discrete	Dynamic	Fingers Hand	No
[19]	2016	RGB-D Camera	-	-	HRI	Gesture Recognition	No	1	Discrete	Static	(Bi) Arm	Yes
						Feature Extraction	No	-	Continuous	Dynamic	Arm	No
[118]	2016	RGB-D Camera	Stereo Infrared Camera	-	HCI	Gesture Recognition	No	25	Discrete	Dynamic	Fingers Hand	No
[119]	2016	RGB Camera	-	-	-	Gesture Classification	No	14	Discrete	Static	Hand Fingers	No
[120]	2017	RGB Camera	-	-	HCI	Gesture Recognition	No	6	Discrete	Static	Fingers	No
						Gesture Recognition	No	1	Discrete	Dynamic	Hand	No
[20]	2017	Accelerometer	-	-	HRI	Feature Extraction	No	-	Continuous	Dynamic	Arm	No
[121]	2017	Epidermal Tactile Sensor	-	-	-	Gesture Classification	No	5	Discrete	Static	Fingers Hand	No
[122]	2017	Infrared Camera	-	-	HRI	Gesture Recognition	No	12	Discrete	Static	Fingers Hand	No
						Gesture Recognition	No	10	Discrete	Dynamic	Hand	No
[62]	2017	Infrared Camera	-	-	-	Gesture Recognition	No	6	Discrete	Dynamic	Fingers Hand	No
[123]	2017	RGB Camera	-	-	-	Gesture Classification	No	24	Discrete	Static	Fingers Hand	No
[124]	2017	RGB Camera	-	-	-	Gesture Classification	No	7	Discrete	Static	Fingers	No
[125]	2017	RGB-D Camera	-	-	-	Feature Extraction	No	-	Continuous	Dynamic	Hand	Yes
[126]	2018	6-Axis IMU	-	-	-	Gesture Recognition	No	9	Discrete	Dynamic	Arm	No
[21]	2018	RGB-D Camera	-	-	HRI	Feature Extraction	No	-	Continuous	Dynamic	Hand	Yes
[11]	2018	RGB Camera	-	-	HCI	Gesture Recognition	No	6	Discrete	Dynamic	Fingers Hand	No
[127]	2018	Infrared Camera	-	-	-	Gesture Classification	No	10	Discrete	Dynamic	Fingers	No
						Gesture Classification	No	26	Discrete	Dynamic	Fingers	No
[128]	2018	RGB Camera	-	-	-	Gesture Recognition	No	5	Discrete	Dynamic	Fingers Arm	No
[129]	2018	RGB-D Camera	-	-	-	Gesture Recognition	No	8	Discrete	Dynamic	(Bi) Hand Arm	No
[61]	2018	Accelerometer	-	-	-	Gesture Recognition	No	6	Discrete	Dynamic	Arm	No
[130]	2019	Accelerometer	Button	-	-	Gesture Recognition	No	10	Discrete	Dynamic	Hand	No
[22]	2019	RGB Camera	-	-	HRI	Gesture Recognition	No	8	Discrete	Static	Fingers	No
[131]	2019	RGB-D Camera	-		-	Gesture Recognition	No	10	Discrete	Dynamic	Hand Arm	No
					-	Gesture Recognition	No	10	Discrete	Dynamic	Hand Arm	No
					HCI	Gesture Recognition	No	10	Discrete	Dynamic	Hand Arm	No
					HCI	Gesture Recognition	No	10	Discrete	Dynamic	Hand Arm	No
					HCI	Gesture Recognition	No	15	Discrete Continuous	Static Dynamic	Fingers Arm	No
[70]	2019	Infrared Camera	-	-	HRI	Feature Extraction	No	-	Continuous	Dynamic	Hand	No
					HCI	Gesture Recognition	No	1	Discrete Continuous	Static Dynamic	Fingers Hand	Yes
[33]	2019	EMG	-	-	HCI	Gesture Recognition	No	4	Discrete	Static	Fingers Hand	No
[79]	2019	Stereo Infrared Camera	-	-	-	Gesture Classification	No	30	Discrete	Static Dynamic	Fingers Hand	No
[132]	2019	Stereo Infrared Camera	-	-	-	Gesture Classification	No	10	Discrete	Static	Fingers	No
[133]	2019	EMG	-	-	-	Gesture Recognition	No	5	Discrete	Dynamic	Fingers Hand	No

2.6 Literature analysis

In Section 2.3.2, we have described how data processing for continuous and discrete gestures implies facing two different problems, i.e., feature extraction and gesture recognition,

respectively. Furthermore, gesture recognition has been organised as a pipeline involving three computational steps, namely pre-processing and feature extraction, gesture detection and gesture classification. In the literature analysis we carried out, this characterisation has been used to identify the specific problems each referenced work aimed at addressing. If we refer to Table 2.2, contributions in the literature focusing on the gesture recognition problem are expected to address all these three computational steps. Instead, the ones addressing gesture classification focus mainly on techniques to classify gestures, although usually they include and employ procedures for pre-processing and feature extraction as well. In all these articles, and often without a clear statement of purpose nor an explicitly stated assumption, the authors rely on the closed-world assumption, i.e., the presumption that the system has complete knowledge about all possible interactions, and whichever action is performed by the user it is part of the gesture dictionary. In order to use these classification techniques in real-world scenarios, the closed-world assumption should be relaxed, and therefore an open-world approach should be considered, i.e., the idea that the system does not have complete knowledge, and therefore actions or gestures performed by the user may be unknown.

A shift towards an open-world assumption is possible if a suitable detection procedure is employed. Although its importance for real-world applications is evident, and notwithstanding the vast attention it received in the context of data-driven approaches, where it is often referred to as *novelty detection* [134], the detection procedure does not attract the interests of researchers interested in gesture interaction, whereas the main focus remains the development of new techniques to solve the classification problem using data-driven approaches. One reason for that lack of attention may be related to the easiness associated with the evaluation of new classification metrics with respect to state-of-the-art approaches, especially when compared to a similar evaluation in case of gesture detection [71]. Therefore, gesture detection approaches are often simplified, i.e., using threshold-based mechanisms, or even naive methods, such as asking users to hold a button while performing a gesture to ease data segmentation. We argue that researchers interested in discrete gestural interaction should prioritise studies focusing on the *overall gesture recognition* and in particular on its *detection* aspects.

It is noteworthy that continuous gestures are under-represented in our survey. This could be the result of a bias in the survey methodology. However, our findings are consistent with the ones of a recent systematic literature review where 89% of the reviewed articles consider discrete gestures [48]. Nonetheless, a number of interesting considerations regarding continuous gestures can be done. Researchers interested in continuous gestures *should not completely disregard* the literature focused on discrete ones since, in order to solve the

gesture recognition problem, feature extraction should be faced as well. In fact, many articles focusing on discrete gestures include an interesting feature extraction analysis, which could be applied to continuous gestures as well [119].

Studies focusing only on continuous gestures are usually limited to reactions that are conceptually simple, e.g., the control of a mobile robot [20], or a *one to one* mapping between the human hand and the robot end-effector for tele-operation purposes [21], where the main problem to solve is geometric in nature. In order to attain more complex reactions, we argue that *a principled integration* between continuous and discrete gestures is necessary. One possible example is a GUI using continuous gestures to move a cursor and adopting at the same time discrete gestures to implement icon selection. Therefore, we believe that novel studies to address the GI-UI problem should focus on the integration of state-of-the-art solutions for continuous and discrete gestures.

Finally, it is remarkable that research on the GI-UI problem mainly focuses on upper limb gestures. This is of course motivated by the intuitiveness in interacting with physical or virtual systems using hands. However, in daily life scenarios, humans extensively use hands to interact with the environment. Therefore, extending gesture-based interaction to other body parts, e.g., feet, could unlock new interaction modalities even in situations whereby hands are already occupied, with obvious positive consequences in case of specialised interfaces for people with special needs.

2.7 Problem relevance

To understand if the problem we are trying to address is relevant we should analyse the distribution of articles over time. Since we did not carry out a systematic review we can not perform a statistical study using our review.

Vuletic et al. 2019 [48] presented a vast literature review for hand gestures used in HCI interfaces including 148 articles from 1980 to 2018. Their review presents a detailed analysis of the distribution of articles over time filtered by type of gesture, application and sensor. However, an analysis of the overall distribution of articles over time is not present. Therefore, we extracted the 148 articles and we have built a graphical representation of their distribution over time. The result, presented in Figure 2.2, highlight that the field starts acquiring popularity in the early 2000s reaching a peak in 2015. Although the analysis highlights that after the 2015 peak the number of publication decreased, the research field seems to maintain its relevance.

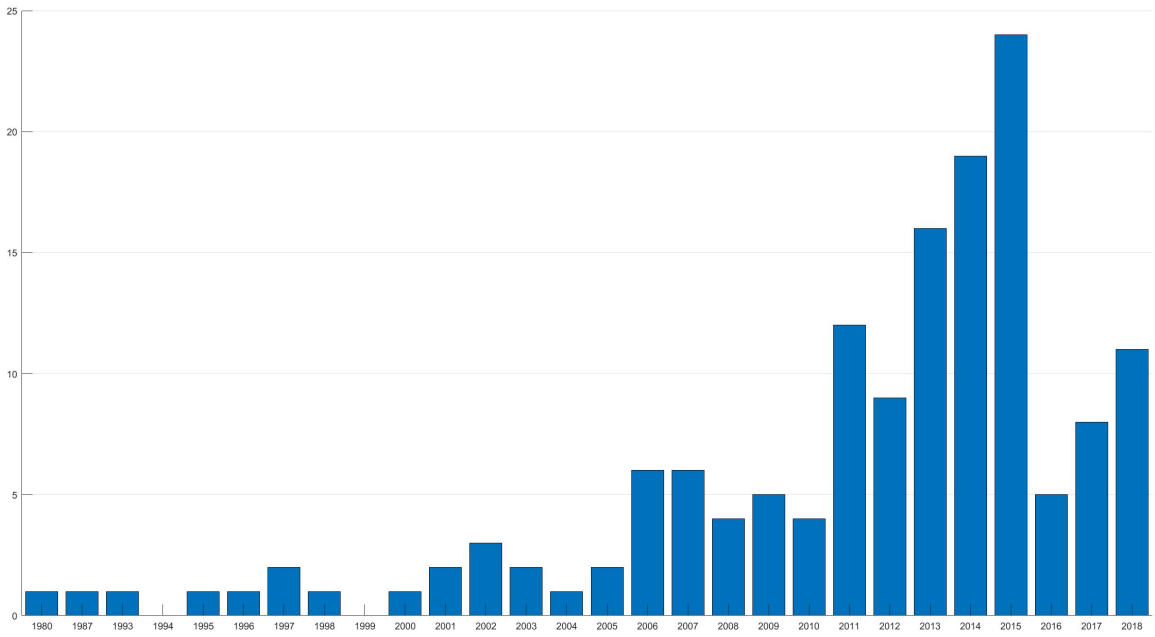


Figure 2.2 Article distribution over time, according to data extracted from Vuletic et al. 2019 [48].

In the original review is proposed an interesting analysis, that we present in Figure 2.3, on the sensing technologies adopted, over time, to perceive gestures. Each box represents the usage of specific sensing technology in one article and the vertical line divides the articles between the one adopting vision-based and wearable technologies, the figure presents a detailed legend as well. From this analysis is evident that the most adopted technologies for gesture sensing are vision-based and really few attention has been directed toward wearable sensing. However market trends, see Figure 2.4, show an increase of adoption of wearable technologies and forecast a further expansion for the forthcoming years. Driving these trends are devices such as fitness trackers and smartwatches, devices not directly projected to support HMI but whose wide adoption can be used to develop new technologies for gestural interaction. Therefore, more attention should be paid to the research on wearable sensing for gestural interaction.

Figure 2.3 Sensing technologies used to implement gesture interaction prototypes, extracted from Vuletic et al. 2019 [48].

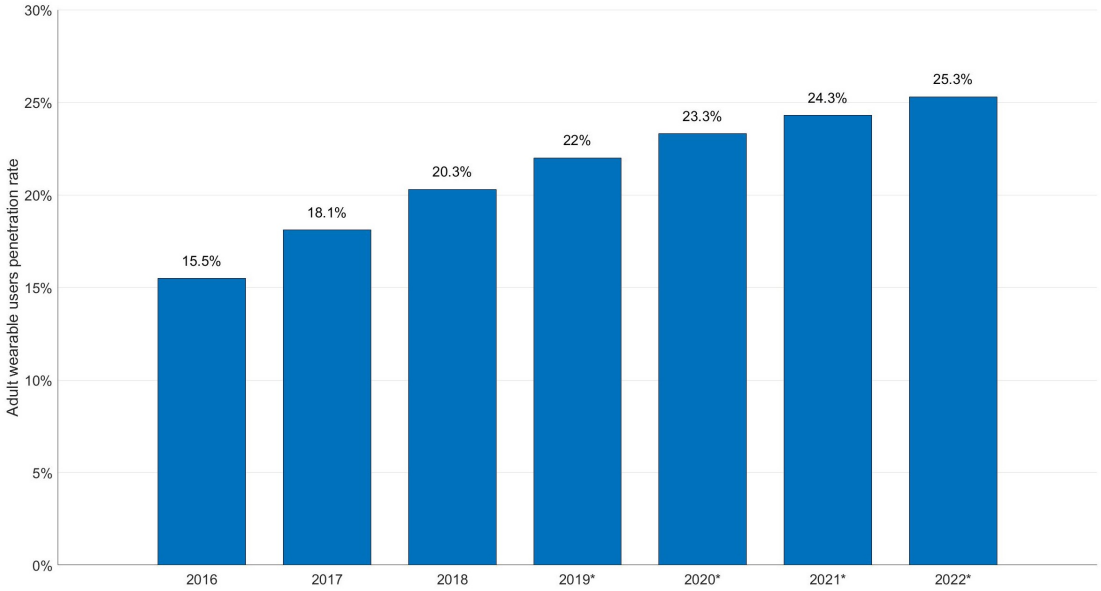


Figure 2.4 Adult wearable users penetration rate in the USA up to 2018 and forecasts up to 2022. Data source from https://www.statista.com/.

Chapter 3

Discrete Gestures

As we have previously seen in Chapter 2, discrete gestures can be used, similarly to a keyboard or buttons, to send commands through an interface to a physical system or a virtual one. An HRI system leveraging gestural interaction requires the design of a gesture dictionary. Previously, we have described the gesture dictionary as a set of matched pairs of commands and their gestural expression. We have even seen how the definition of the dictionary is tightened to the satisfaction of human-related constraint and how it can influence the solution to the problem statement. What differentiates the problem statement for discrete gestures is the data processing since after the data stream has been processed (or not in case we directly use raw data) to extract features the data stream should be segmented and classified. Since we start from scratch in the designing of our interface we can decide from where we would like to start, we should solve different problems such as dictionary design, choice of sensors, develop the processing procedure and design the reaction of the system. All these elements are not completely disjunct and can be approached in different orders.

3.1 Preliminary Considerations

In this Chapter, we focus on a specialization of the problem (Discrete GI-UI), by considering the case where robot modalities have to be selected using a gesture-based interface leveraging discrete gestures. Collaborative robots (CoBot) can cooperate closely with the human operator thanks to different functionalities that can be activated autonomously or recalled by the user. Therefore, the gesture-based interface should let the user select the different functionalities intuitively. The association of each functionality to a specific gesture is not ideal since an increase of the functionalities is going to increase the lexicon size, leading to lower learnability and higher mental load, as we have seen in Section 2.4. An approach to

have an intuitive interface while maintaining constant the number of gesture that you need to navigate the interface is the adoption of a menu-based interface [135]. The intuitiveness of such an interface is due to its wide adoption in everyday technology. Smartphone apps are often organized following this paradigm with a vertical menu explored through finger swipes and even the Google search result page is organized in a similar way. However, these are examples in which the technology allows for a continuous interaction while we are facing the Discrete GI-UI problem. Our case of study is more similar to the classical BIOS menu through which you can navigate using keyboard arrows.

The choice of a menu-based approach delimits the size of our gesture dictionary. We suppose a menu in which it is possible to move only in one direction, horizontal or vertical. Therefore, gestures should have the same functionality as arrow keys. Furthermore, we need one gesture to select an option and another to go back to the previous menu. Overall we need a minimum of four gestures but we could want to have some more gestures available to implement shortcuts. However, before designing the gestures we have to decide how we are going to sense them since depending on the sensor choice, not all the gestures could be easy to recognize.

The scenario in which we want our interface to operate is an industrial one in which the user collaborates with CoBots both my mean of our interface and physical contact. As we have observed in Chapter 2 gesture sensing can be classified in image bases and non-image based, and from table 2.2 we can notice the most common non-image based approach is the usage of accelerometers or inertial measurement units (IMUs). This kind of sensors should be worn by the user, are cheap and their data stream is easy to process and transmit. The user wearing such a sensor could easily change the working station without having to worry about anything since he is carrying with him the sensing equipment. On the contrary, cameras are usually more expensive, each working station should have at least one camera to allow the user to use on different robots the gesture-based interface, their data are heavier to be processed and the accuracy of the gesture recognition can be affected by lighting condition and occlusions. This last element is particularly relevant in our scenario since the human should work in close contact with the robot and this is going to cause frequent occlusions. An alternative for accelerometers is represented by Electromyography (EMG) sensors [133]. However, they are more expensive then accelerometers, their data is more difficult to be interpreted and easily wearable EMG devices are bulky respect to normal wearables. On the other side, the compactness of IMU data, that makes them easy to be transferred and processed, implies a simpler data source that can not depict semantic information with a detail level similar to sensors such as cameras. For this reason, in some scenarios, IMU data

can result ambiguous and difficult to be interpreted. However, IMUs have been proved to be effective for gesture sensing, therefore, we opted out for using accelerometers that can be easily embedded in wearables such as smartwatches and wristbands. An accelerometer worn at the wrist forces the gesture focus. Therefore we are going to consider discrete arm gestures. For this kind of application, we do not need a spatially related gesture and dynamic gestures are more advisable. We do not know, and we do not want to, the kind of work the user should do in collaboration with the robot. It could imply few movements and lots of static postures or few postures and lots of movements. In the first case, static gestures would be more easily confused with normal operations while in the second one dynamic gestures will be the one confused. Anyway, since we are not going to make any assumption on the collaborative task the safest thing to do is to choose the kind of gesture that allows you to create the most distinct gestures. With dynamic gestures we can define specific trajectories and, while taking into account the user comfort, we can build a dictionary reasonably unique respect to the usual human motions. Therefore, for our application, we are going to use discrete dynamic spatial unrelated arm gestures. In order to perform some preliminary experiments, we have designed a mocap dictionary composed of six gestures. This dictionary is going to be presented later in this chapter.

3.2 Gesture Recognition - Background

Gesture recognition, as we have previously described, is the process that leads to the identification and classification of a temporal sequence from a continuous data stream. Although we are focusing on this specific problem, we can not ignore solutions to a similar problem such as the activity recognition one. Activities such as walking, running or drinking differ from gestures since are not meant to actively convey information. Nevertheless, many of the sensors and techniques used for activity recognition are in overlap with the ones used for gesture recognition. Therefore, in the following section, we are going to review even articles from this research field.

Since we have determined the sensor we want to use and the kind of gesture we are going to consider we can have a more clear idea on how to solve the gesture recognition problem and which is the state-of-the-art for this particular instance of the gesture recognition problem. Table 3.1 groups up all the works that in the previous section we have found out were using an accelerometer, 6-axis IMU (accelerometer and gyroscope) or 9-axis IMU (accelerometer, gyroscope and magnetometer) to perceive discrete dynamic gestures. We have discarded all the works that integrate IMUs with other sensors and in particular, all the works using a

physical [72, 50, 85, 86, 88, 100, 107, 113, 115, 130] or a virtual [94] button to detect the intention of the user to perform gestures, since we do not want to overload the operator and we aim to a more intuitive interaction. Furthermore, only the works solving the complete gesture recognition problem are considered.

Table 3.1 Summary of literature for discrete gestures perceived using IMUs.

Article info		Sensing	Reaction	Gestures		Frequency	(0) Preprocessing	(1) Feature Extraction	(2) Detection	(3) Classification	Order			
Ref	Year			Size	Focus									
[16]	2010	Accelerometer	HRI	6	Arm	1600 Hz	Yes	No	Threshold	DTW	2	0	3	-
[60]	2015	Accelerometer	-	12	Fingers	50 Hz	Yes	Yes	Threshold	Threshold	0	2	1	3
[106]	2015	9-Axis IMU	-	10 26 8	Hand Hand Hand	75	Yes	Yes	Threshold	DTW	0	2	1	3
[112]	2016	Accelerometer	-	5 5	Arm Arm	-	Yes	Yes	Threshold	DTW	0	1	2	3
[114]	2016	6-Axis IMU	-	5	Fingers	50	Yes	Yes	DTW-k-NN + Threshold	SVM Naive Bayes Logistic Regression k-NN	0	1	2	3
[116]	2016	6-Axis IMU	HCI	6	Hand	-	Yes	Yes	Threshold	DTW	0	1	2	3
[126]	2018	6-Axis IMU	-	9	Arm	-	Yes	No	Threshold	GRU NN	2	0	3	-

Since none of the selected works considers user-defined gestures or spatial one and all solve the gesture recognition problem for discrete dynamic gestures, to maintain a compact representation we have removed the processing, the temporal and the effect columns. Furthermore, we added some columns to describe the approach used to solve the gesture recognition problem: frequency of data acquisition, preprocessing presence, feature extraction presence, detection technique, classification technique and order in which the four phases are executed.

Preprocessing Accelerometer data are typically affected by high-frequency noise that can be filtered out using different techniques such as *moving average filters* [60, 115, 106, 136] [112], *median filters* [137], *temporal compression* [88], *quantization* [87] or *Hanning filters* [16]. Accelerometers measure the proper acceleration of the object they are attached to, which includes the gravity acceleration and any other acceleration that the object is subject to (in the case of wearable sensors, any other acceleration produced by a person's movements). The gravity acceleration can be used as an independent source of information for the classification [137], to isolate body acceleration [136] or to compute the arm orientation [112]. The preprocessing phase is typically devoted to noise filtering and to the separation of gravity and body acceleration components. The latter procedure typically involves the use of a *low-pass filter* [136] [137].

Feature Extraction Acceleration data recorded during the execution of a gesture appear as time series. In order to reduce the complexity of the classification problem some solutions suggest extracting discrete features using *statistical analysis* [115, 114], the *Haar Transform* [92] or extraction of the parameters from an *autoregressive model* [136]. Alternatively, it is possible to extract continuous features such as sensor orientation [138], jerk [60] or accelerometer magnitude [16].

Detection Whenever the processing of the acceleration data is expected to be done online, the problem of recognizing gestures should encompass their detection. The accelerometer time series should be segmented to isolate the portion of data where a gesture is detected. Simple segmentation approaches require the end-user to communicate through buttons [115, 88, 87] or touch-screens [94] when a gesture starts and ends. More advanced approaches perform direct, when the detection is performed on the sensing data, or indirect detection, when it is performed on the classification results [63]. Direct detection typically focuses on detecting variations in the data stream [115, 112, 16] to identify the gesture end and start points. This process induces a sporadic gesture recognition whose main limitation is that the gesture must necessarily finish before the classification process starts. Alternatively, indirect detection can be achieved using a moving horizon window combined with a threshold mechanism to discriminate between unknown and known activities [137] and gestures [114]. At each shift of the sliding window the contained data are processed by a classifier returning, together with the label, a distance metric that is used to determine whether the gesture occurred or not.

Classification Discrete features can be used to classify gestures using approaches based on *Feed Forward Neural Networks* (FNN) [115] [136] or *Support Vector Machines* (SVMs) [92, 114]. An alternative approach envisions the use of continuous features to build time-dependent models, for example, using *Gaussian Mixture Modelling* (GMM) and *Gaussian Mixture Regression* (GMR) [137] or to simply store them as templates [112, 88, 87, 16, 94], and then use techniques such as *Dynamic Time Warping* (DTW) [112, 16, 138] or *k-Nearest Neighbours* (k-NN) to compare them with the data that should be classified. DTW is a *de facto standard* solution in the literature, possibly combined with other methods such as affinity propagation [88] and template adaptation [87]. Adopted alternatives to DTW are represented by *Mahalanobis distance* [137], *Global Alignment Kernel* [94], *Recurrent Neural Network* (RNN) classifiers [139] and *Neural Networks composed of convolutional layers and gated recurrent unit* (GRU-NN)[126].

Order As we already highlighted in the previous chapter, we can notice that does not exist a specific order in which the data processing should be performed, although we can find some patterns. Preprocessing is always executed before extracting features, if any feature is extracted, and classification is performed after the feature extraction. This is due to the fact that the feature extraction, in the gesture recognition process, is performed to facilitate the classification.

It is important to highlight that almost all the work reported in Table 3.1 performs direct detection. This implies that, although it is not clearly stated in the articles, the gesture recognition problem is solved under the close-world assumption, i.e. the presumption that the system has complete knowledge and whichever action performed by the user is in the gesture dictionary, and not under the open-world assumption, i.e. the presumption that the system does not have complete knowledge and therefore, actions or gestures performed by the user can be unknown. Of course, this does not consider unintentional movements or other kinds of motions that the user does not want to be interpreted as gestures. The only exception to this is Wen et all. 2016 [114] that overcomes the close-world assumption developing an algorithm that leverage DTW and k-NN to compute the distance of the observed sequence, stored in a sliding window, with gestures in the dictionary. If the distances do not overcome a certain threshold the sequence is discarded and it is not processed by the classifier. We observe that the only work overcoming the close-world assumption is doing it by introducing in the detection process a metric of distance respect to the gesture classes. All the works using DTW computes the distances of the selected sequence with al the templates in the gesture dictionary, then they classify the gesture with the label of the closer template. In this work, instead, they use these distances to determine if a gesture is present or not and then they call another classifier. Basically, this is an indirect detection, since the data used to perform the detection are not the extracted features or the plain data but the result of a process that normally could be directly used to perform the classification. To solve the close-world assumption the system should implement a novelty detection algorithm, the mechanism by which the system can understand that a sequence is unknown. We introduced the novelty detection problem for completeness although we are not directly addressing it because of its extent, for an extensive review refer to Pimentel et al. 2014 [134].

We think that the usage of indirect detection can be a simple and effective approach to solve the gesture recognition problem in an open-world scenario. As we have seen thresholds are a common approach to perform the gesture detection. If we want to use a threshold mechanism our classification module, together with the classification label, should output something like a similarity metrics, such as for DTW, or a classification probability. Recurrent

neural networks can be used to implement a probabilistic classifier [139] whose output can be used to feed an indirect detection procedure. Therefore, we are going to investigate the integration of an RNN probabilistic classifier in an architecture that uses a moving horizon window, as in [137], to ensure on-line, *as-early-as-possible*, gesture recognition. Specifically, our recognition procedure, that we refer to as SLOTH, uses raw inertial data and relies on a novel mechanism, which models gestures occurrences on top of neural network output patterns, for discriminating between known and unknown gestures.

3.3 Gesture Recognition - SLOTH

The following sections are extracted from Carfì et al. 2018 [27].

3.3.1 System's Architecture

SLOTH processes data collected by a tri-axial accelerometer worn by users on their right wrist and, whenever a discrete arm gesture is recognized, it returns a label. As described in Figure 3.1, the overall architecture is composed of three modules: *Data Feeding*, *Recurrent Neural Network* (*RNN*) and *Detection*.

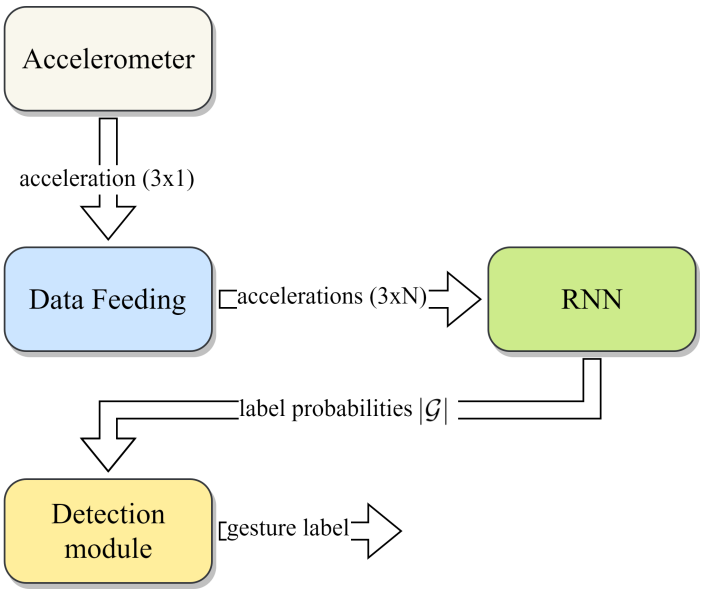


Figure 3.1 SLOTH's architecture.

Data Feeding

The *Data Feeding* module receives raw acceleration data from a tri-axial accelerometer at a fixed frequency, f, and stores them in a buffer of size N, where N depends on gestures length. Once the buffer is full (i.e., after N time instants), the *Data Feeding* module sends the content of the buffer to the RNN module. At each new sample, the content of the buffer is shifted forward to include the new sample and the updated buffer content is sent to the RNN. The *Data Feeding* module does not introduce time steps delays.

Recurrent Neural Network

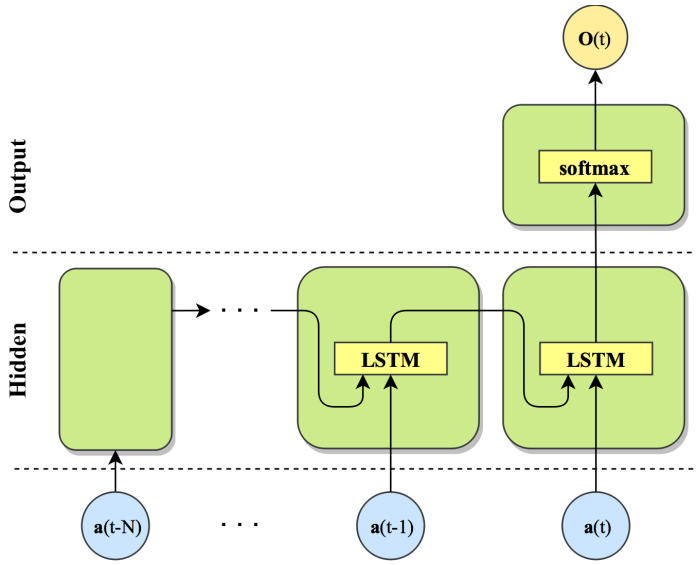


Figure 3.2 The unfolded computational graph of the RNN. The RNN is composed of an LSTM recurrent layer to model temporal relations and a softmax layer for classification. The network receives as input a tri-axial acceleration time series and outputs label confidences.

An RNN, structured as in Figure 3.2, has been chosen for its capability to model time-dependent behaviours to perform a probabilistic classification of gestures using time series of tri-axial linear accelerations. The RNN receives as input a time series $\boldsymbol{a}(t) = [a_x(t), a_y(t), a_z(t)]$. The input is fed to a Long Short-Term Memory (LSTM) hidden layer, which learns long-term temporal dependencies. The output of the hidden layer, for the last input time step, is fed to a *softmax* output layer that returns the probabilities for $\boldsymbol{a}(t)$ belonging to each considered gesture. The network, working under the close-world assumption, discriminates between gesture classes described in the dictionary:

$$\mathscr{G} = \{G_1, \ldots, G_{|\mathscr{G}|}\}, \tag{3.1}$$

Each gesture class G_i is assumed to be *unique* (i.e., a data stream cannot be classified as an instance of different gesture classes at the same time) and *independent* (i.e., each class is not related to, as a component or sub-part of, other classes), and characterized by an average temporal duration S_i. As described in Figure 3.1 the RNN receives as input a time series $\boldsymbol{a}(t)$ of dimension $3 \times N$ with

$$N = \max_{i=1,\ldots,|\mathscr{G}|}(S_i) \tag{3.2}$$

Since the network has been trained over $|\mathscr{G}|$ gesture classes, the output vector $\boldsymbol{o}(t)$ has dimension $|\mathscr{G}|$. During the training of the RNN, beside acceleration samples from $g_j \in G_j$, a target vector $\boldsymbol{v}$ for $\boldsymbol{o}(t)$ is given, normalized such that all values are zero except for $v_{G_j} = 1$. Therefore, when the trained network is used, each element $o_i \in [0,1]$ and, when acceleration data from g_j are given as input to the network, o_j tends to one while others tend to zero.

Detection

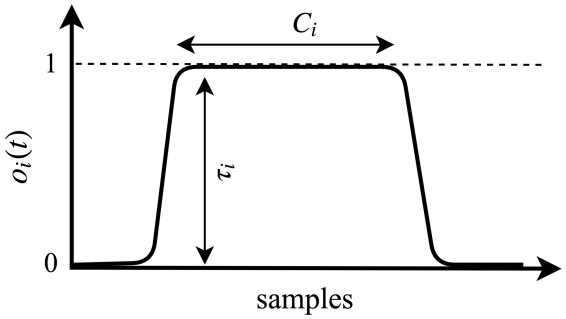


Figure 3.3 Plateau behaviour expected in $o_i(t)$ when gesture G_i occurs.

The detection module receives the neural network output $\boldsymbol{o}(t)$ representing probabilities associated with each gesture class. The detection module processes the stream of gesture probabilities to detect and classify known gestures, relaxing the close-word assumption introduced by the RNN.

As described in Sec. 3.3.1 the neural network reacts at time t to gesture g_i by raising $o_i(t)$ to 1. This implies a positive peak in the derivative:

$$\Delta o_i(t) = o_i(t) - o_i(t-1). \tag{3.3}$$

Since it is the derivative of $o_i(t)$, $\Delta o_i(t)$ is a scalar of value in the interval $[0,1]$. We define the peak instant t_p for the gesture g_i as the time instant for which $\Delta o_i(t_p) > \rho$. The threshold ρ allows for filtering out small fluctuations due to noise.

The network is trained with many g_i examples that differ in time length and signal magnitude, therefore the resulting network is able to recognize temporal pattern associated with G_i in different conditions. Furthermore, since the considered gestures are unique and independent (Sec. 3.3.1), their temporal patterns are unique and independent as well and the network needs to process only a portion of the gesture before being able to classify it. For these reasons and because of the buffering mechanism, the expected $o_i(t)$ behaviour when g_i occurs is represented by a plateau as in Figure 3.3.

Due to the buffering performed by the *Data Feeding* module every sample $\boldsymbol{a}(t)$ with $t > N$ is processed N times. Assuming as a classification limit case the presence in the buffer, as first or last element, one sample $\boldsymbol{a}(t) \in g_i$, the previously described model can be formalized as:

$$\lim_{\eta_i \to 1} \left(\eta_i - \frac{1}{C_i} \sum_{t_p}^{t_p+C_i} o_i(t) \right) = 0^-, \tag{3.4}$$

where

$$\frac{1}{C_i} \sum_{t_p}^{t_p+C_i} o_i(t) = A_i \leq 1, \tag{3.5}$$

and $C_i < S_i + N$. In particular, Eq. 3.4 describes the plateau behaviour of $o_i(t)$, while Eq. 4.5 describes the limit case, when the network classified perfectly G_i for C_i samples, then $A_i = 1$. The described model implies that when g_i occurs, then $A_i \geq \eta_i$. Note that in Eq. 3.4 η and C are presented as gesture dependent parameters, in fact ideally the network response should be homogeneous for all the gesture classes but this does not typically happen, thus η_i and C_i should be defined experimentally.

Iteratively and independently for all the gesture classes in $\mathscr{G}$, the detection module:

- identifies positive peaks (*detection*);
- classifies the samples in the input buffer as an occurrence of gesture class G_i if the condition $A_i \geq \eta_i$ is satisfied (*classification*).

Different buffer shifts containing samples referring to a single g_i could satisfy the condition $A_i \geq \eta_i$. Therefore, in order to avoid g_i to be recognized multiple times, each positive peak is associated with only one recognition.

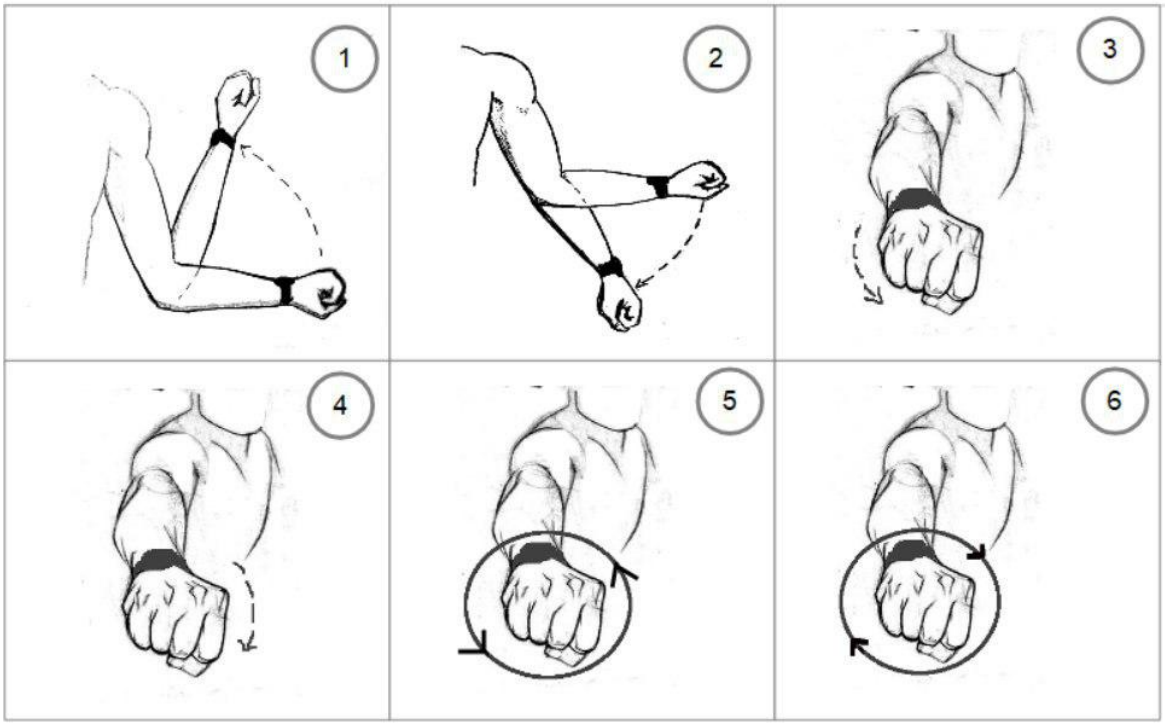


Figure 3.4 Visual representation of the gestures composing the dictionary.

3.3.2 Dataset

Experiments to acquire right wrist acceleration data are performed using an LG G Watch R smartwatch. The smartwatch is equipped with a tri-axial accelerometer and it is paired with a smart-phone that receives the data and saves them on file. The system collects data at a frequency $f = 40\ Hz$, this data are then downsampled at 10 Hz. The gesture dictionary is composed of the six gestures represented in Figure 3.4. All gestures assume the same starting pose for the arm: the elbow bent at 90 degrees while in contact with the flank and the hand held horizontally and pointing forward. Similarly, all gestures end when the arm is back in the starting pose. As described in Figure 3.4, in G_1 the arm moves upward maintaining fixed the elbow and with no wrist twist (Figure 3.5a), in G_2 the arm moves downward maintaining fixed the elbow and with no wrist twist, in G_3 the arm is stretched and the wrist twists clockwise, in G_4 the arm is stretched and the wrist twists counter-clockwise (Figure 3.5b), in G_5 the hand performs a clockwise circle with no wrist twist and, lastly, whereas in G_6 the hand performs an anti-clockwise circle with no wrist twist. Using the afore-described equipment, we collected two datasets of the six gestures composing the vocabulary, which we refer to as *Dataset A* and *Dataset B*:

Dataset A is used to train and test the RNN module. Ten volunteers performed nine times the six gestures described above, providing a total of 540 sequences. These sequences are

(a) Volunteers performing gesture G_1.

(b) Volunteers performing gesture G_4.

Figure 3.5 Gestures examples performed by two volunteers wearing the smartwatch on their right arm.

manually cut so that they only contain acceleration samples which refer to the execution of the gestures. The dataset has been divided, preserving the balance of volunteers and gestures, in two subsets, respectively used for the training (70%) and testing (30%) of the RNN module.

Dataset B includes 15 sequences collected from one volunteer, known by the system. While in *Dataset A* one sequence refers to one execution of one gesture, sequences in *Dataset B* contains from a minimum of 6 to a maximum of 12 gestures (providing approximately 20 executions per gesture), separated by a non-constant number of samples in which the user remains in the starting pose. There are no consecutive executions of the same gesture in the sequences. The dataset is manually tagged.

3.3.3 Implementation

The modelling and recognition system presented above has been implemented in MATLAB R2017b.

Data Feeding. In the tests with sequences of *Dataset B*, the *Data Feeding* module is in charge of simulating the online usage of the architecture. Given a sequence belonging to *Dataset B*, it loads the acceleration data sample by sample and feeds them to the RNN module through the buffer, whose size has been set to $N = 40$ samples.

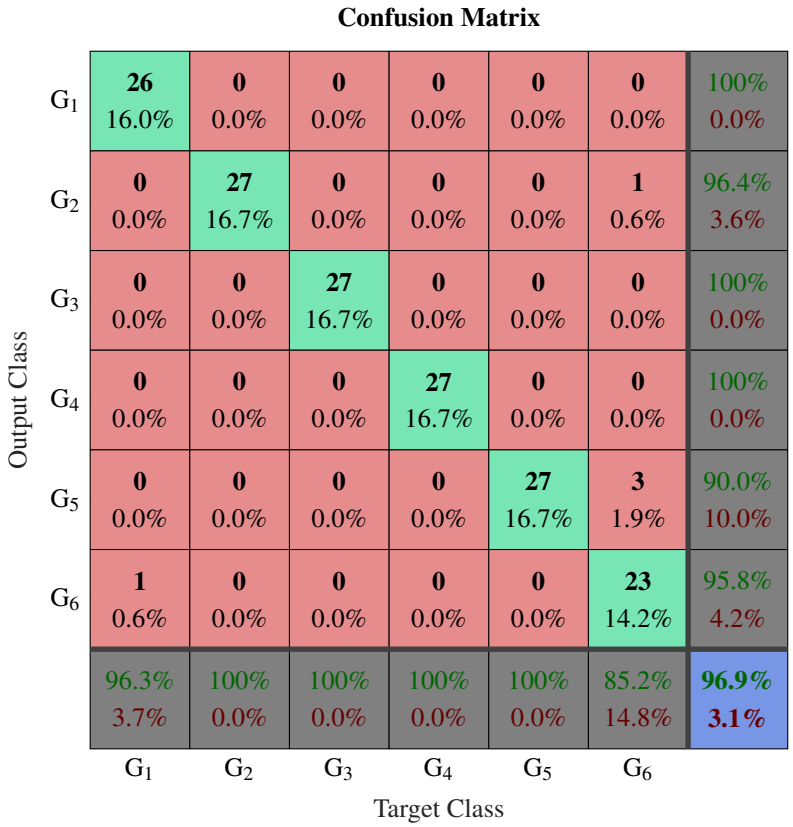


Figure 3.6 Confusion matrix for the RNN offline testing (*Dataset A*). The bottom row reports the *recall* measures while the rightmost column reports the *precision* measures. The blue cell reports the overall accuracy.

Recurrent Neural Network. As shown in Figure 3.2, the RNN is composed of an LSTM layer and a *softmax* layer, which are implemented using standard MATLAB libraries. The hidden layer is composed of 32 neurons, and the training procedure uses the *cross entropy* loss function and *stochastic gradient descendent with momentum* as an optimizer. Since the results of Eq. 3.2 for *Dataset A* is $N = 40$, the input size of the network is 3×40. Therefore the training sequences $\boldsymbol{a}(t)$ containing less than 40 samples are padded with $\boldsymbol{a}(1)$ at the beginning. During the training and the offline testing phases, the buffer mechanism is not present and for each sequence $\boldsymbol{a}(t)$ the network returns a single vector $\boldsymbol{O}$. Since the selected gesture dictionary has dimension $\mathscr{G} = 6$, the size of $\boldsymbol{O}$ is 6 as well. The network output $\boldsymbol{O}(k)$ for each sequence k in the test set, containing g_i, is processed by an *argmax* function to determine the i-label. Figure 3.6 shows the confusion matrix obtained by the RNN on the testing *Dataset A*. It can be seen that the RNN achieves good results in terms of accuracy, precision and recall.

Continuous Gesture Recognition. The GCR module has three parameters, $\boldsymbol{C}$, $\boldsymbol{\eta}$ and ρ, which must be set according to the gesture dictionary and to the neural network response. In order to filter out only small fluctuations, in the interval of possible values $[0,1]$, it is picked $\rho = 0.2$. Instead, $\boldsymbol{C}$ and $\boldsymbol{\eta}$ can be defined as:

$$\begin{aligned} \boldsymbol{C} &= \alpha(\boldsymbol{S}+N), & 0 < \alpha \leq 1, \\ \boldsymbol{\eta} &= \gamma \boldsymbol{M}, & 0 < \gamma \leq 1, \end{aligned} \tag{3.6}$$

where $\boldsymbol{M}$ is the average network response for each gesture, such that

$$M_i = \frac{1}{n_i} \sum_{k=1}^{n_i} O_i(k) \tag{3.7}$$

given that n_i is the number of sequences contained in the dataset referring to G_i. From an analysis of *Dataset A*, it results:

$$\begin{aligned} \boldsymbol{S} &= [38,38,37,37,38,38], \\ \boldsymbol{M} &= [0.996,0.996,0.97,0.995,0.934,0.917]. \end{aligned} \tag{3.8}$$

Setting $\alpha = 0.25$ and $\gamma = 0.9$ leads to:

$$\begin{aligned} \rho &= 0.2, \\ \boldsymbol{C} &= [20,20,19,19,20,20], \\ \boldsymbol{\eta} &= [0.896,0.896,0.873,0.895,0.84,0.825] \end{aligned} \tag{3.9}$$

as final set of parameters.

3.3.4 Experimental Evaluation

Figure 3.7 shows the confusion matrix obtained by testing SLOTH, in the implementation presented above, with the sequences of *Dataset B*. The detection module presented in Sec. 3.3.1 allows for relaxing the close-word assumption, which is represented in Figure 3.7 using the tag "N. G." (Not a Gesture). In the figure, it is possible to observe that, the precision is very high for all gestures (the minimum is 94.4% for G_5), while the recall is lower, especially for gestures G_3 (55%) and G_6 (45.5%). In both cases, most of the misclassified executions are not recognized at all (N.G.). Figure 3.8a presents the recognition results and the timings for one continuous sequence included in *Dataset B* which contains each gesture twice (specifically, in the order $G_1, G_2, G_1, G_2, G_3, G_4, G_3, G_4, G_5, G_6, G_5, G_6$). The three graphs in

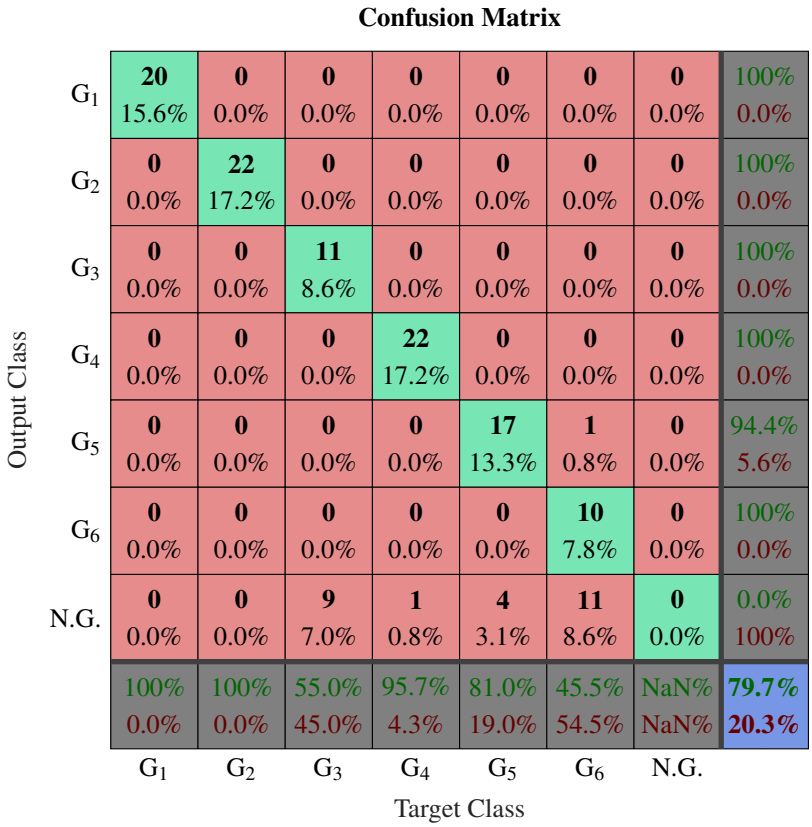
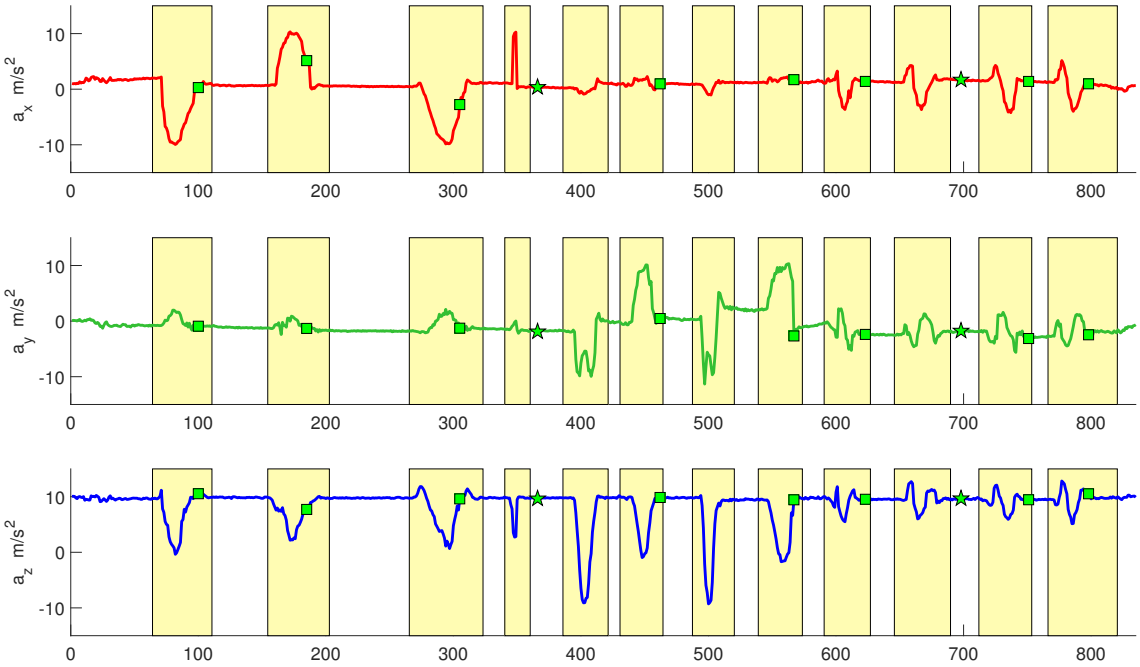


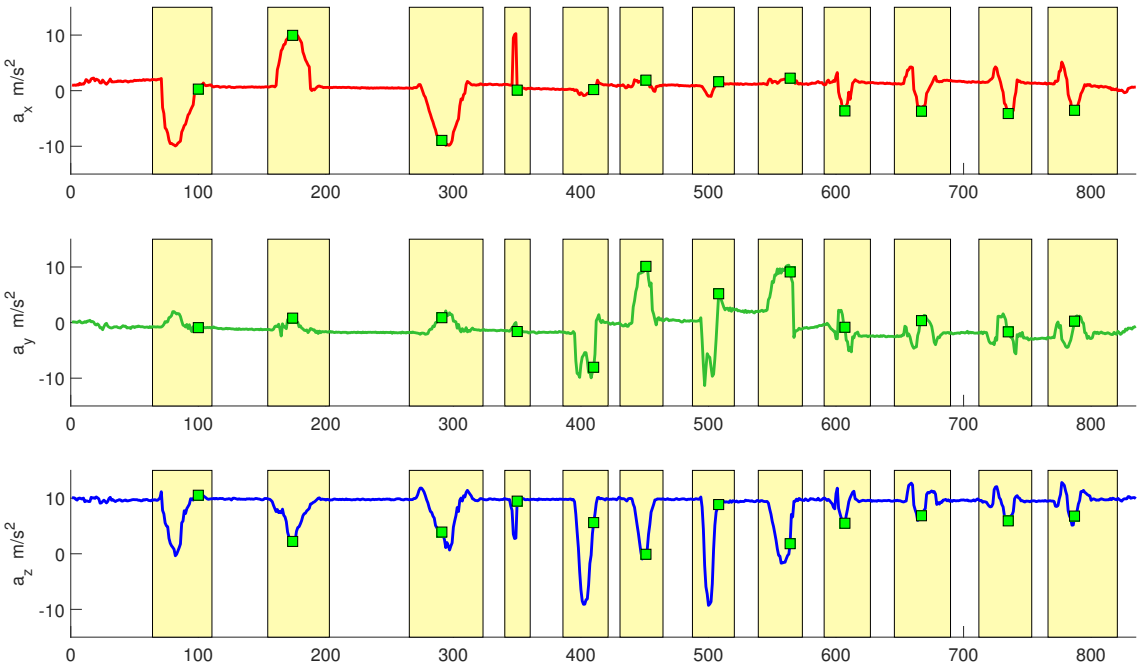
Figure 3.7 Confusion matrix for the online testing (*Dataset B*) with parameters from Eq. 3.9. The bottom row reports the *recall* measures while the rightmost column reports the *precision* measures. The blue cell reports the overall accuracy.

Figure 3.8a show, from top to bottom, the x, y and z acceleration components. Yellow boxes denote gesture instances, while green squares and stars denote correct classifications. More precisely, green squares indicate when the recognition occurs before the end of the gesture while green stars denote when the recognition occurs after the end of the gesture. As Figure 3.8a shows, out of the 12 gestures contained in that recording, 10 are correctly classified and before their end, 2 are correctly classified after their end and 2 are not classified.

The tests on *Dataset B* reported in Figure 3.6 and Figure 3.8a, show that the parameter settings discussed in Sec. 3.3.3 are very conservative, giving clear preference to precision over recall. This behaviour is well suited for applications where gestures are used to control a robot, for example, but it may not be desirable in other contexts. Parameters $\boldsymbol{\eta}$ and $\boldsymbol{C}$ allow for controlling this behaviour. In particular, increasing these values makes the expected plateau longer ($\boldsymbol{C}$) and higher ($\boldsymbol{\eta}$), thus increasing precision, while reducing them makes the expected plateau shorter ($\boldsymbol{C}$) and lower ($\boldsymbol{\eta}$), thus increasing the recall. Furthermore reducing $\boldsymbol{C}$ allows for recognizing gestures earlier, thereby increasing the reactivity of the system.



(a) Classification output for an online test (*Dataset B*) with system parameters from Eq. 3.9.



(b) Classification output for an online test with $\alpha = 0.05$ and $\gamma = 0.9$

Figure 3.8 From top to bottom are represented the x, y and z acceleration components. Yellow boxes denote gestures instances, while green squares and stars denote correct classifications. Green squares denote when the classification occurred before the end of the gesture and green stars denote when the classification occurred after the end of the gesture.

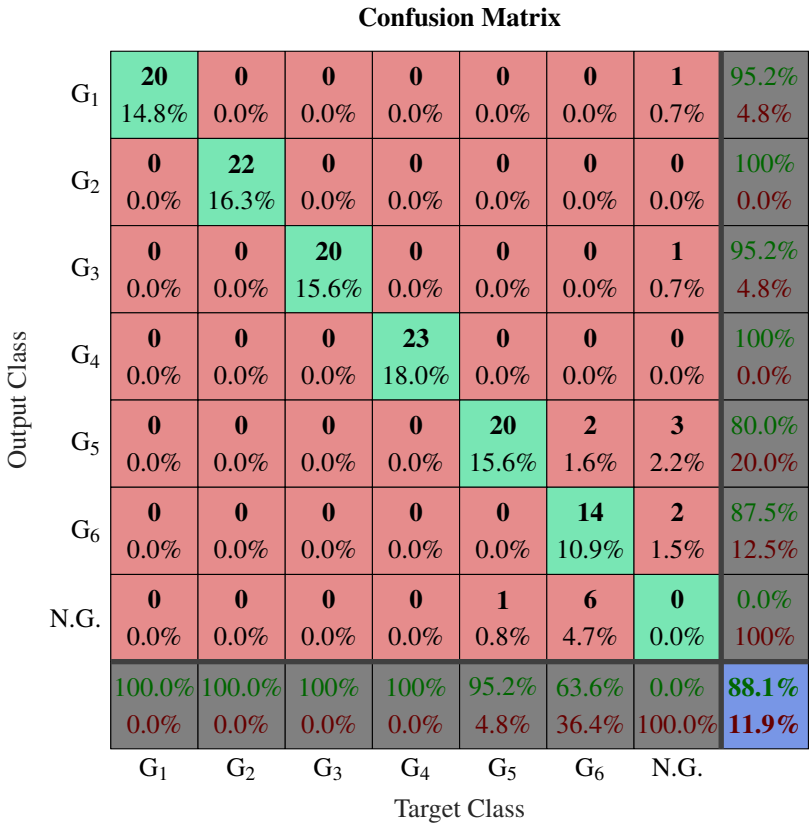


Figure 3.9 Confusion matrix for the online testing (*Dataset B*) with $\alpha = 0.05$ and $\gamma = 0.9$. The bottom row reports the *recall* measures while the rightmost column reports the *precision* measures. The blue cell reports the overall accuracy.

To verify whether and to what extent the above statement holds, we have repeated the tests on *Dataset B* two more times, once decreasing $\boldsymbol{C}$ to $\alpha = 0.05$ while keeping $\boldsymbol{\eta}$ to the value defined in Eq. 3.9, and one decreasing $\boldsymbol{\eta}$ to $\gamma = 0.6$ while keeping $\boldsymbol{C}$ as defined in Eq. 3.9. The results of the first test are shown in Figure 3.9 and Figure 3.8b, while the results of the second test are shown in Figure 3.10.

Figure 3.9 shows that, as expected, new $\boldsymbol{C}$ values yield an increase in the recall at the expenses of a small decrease in precision. Moreover, the number of samples required to issue the label (see Figure 3.8b) is significantly smaller than that with the values defined in Eq. 3.9. Similarly, Figure 3.10 shows that new $\boldsymbol{\eta}$ values yield an increase in the recall at the expenses of a small decrease in precision.

All the performed tests as well as the RNN offline testing presented in Figure 3.6 highlight difficulty in classifying of G_6. This is probably a consequence of using raw acceleration data, which include a component related to gravity and one, in our case, related to the person's arm movements. When a person performs gestures G_1, G_2, G_3 or G_4, the gravity component

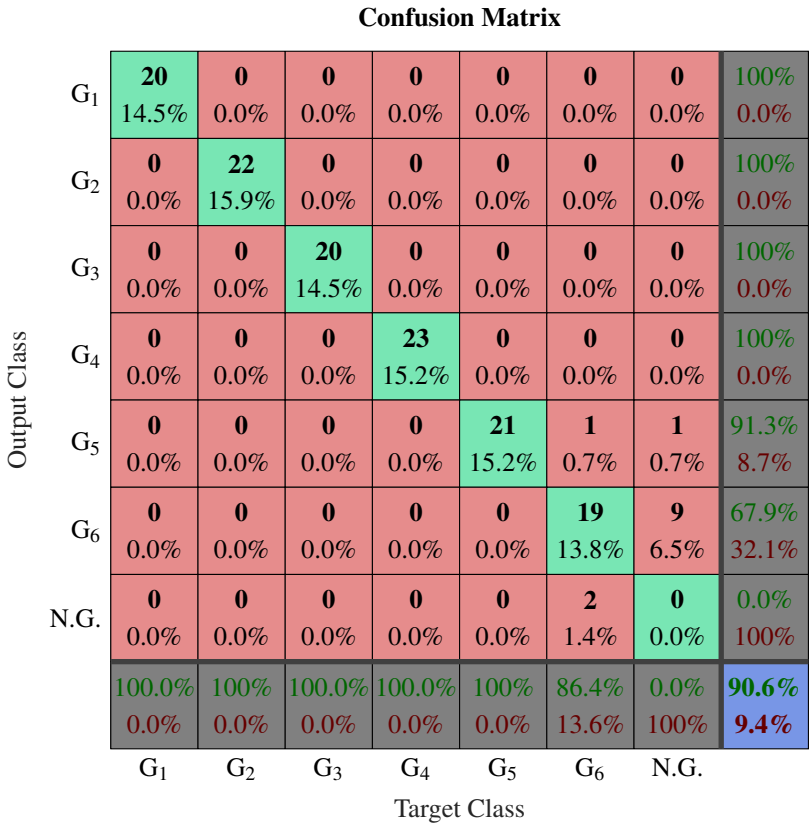


Figure 3.10 Confusion matrix for the online testing (*Dataset B*) with $\alpha = 0.25$ and $\gamma = 0.6$. The bottom row reports the *recall* measures while the rightmost column report the *precision* measures. The blue cell reports the overall accuracy.

shifts from one accelerometer axis to another, thus ensuring that the acceleration patterns encode sensible variations. This does not happen in the case of gestures G_5 and G_6. Since the gravity acceleration is by far the most prominent acceleration component, we argue that its shift between accelerometer axes helps the classification and, as a consequence, its absence causes the performance loss.

It is worth noticing that even in the configurations prioritizing recall over precision, precision remains very high, thus proving the robustness of the proposed approach. The main drawback of our approach is that the RNN needs to be retrained every time a gesture is added/deleted, and therefore the system's performance depends on the chosen combination of gestures.

3.4 Gesture Recognition - Comparative Study

SLOTH achieves good results in on-line gesture recognition overcoming the close-world assumption thanks to a sliding window mechanism and indirect detection, but it lacks in modularity since every change in the gesture dictionary implicates a retrain of the RNN.

In Section 3.2, we have described the importance of solving the gesture recognition under the open-world assumption and we have described why indirect detection allows to do so. Two similar approaches using indirect detection to solve the gesture [114] and activity [137] recognition problem have been reviewed. The former makes uses of DTW and k-NN while the latter uses GMM and GMR to build activity templates and Mahalanobis distance to compute the distance between the incoming data and the activity in the dictionary.

Since they both solve the same problem that SLOTH does, we think that a comparative study is necessary to have a clear idea of how good SLOTH recognition results are. The original article does not present enough information for the DTW and k-NN approach. For these reasons we have decided to compare SLOTH with the method leveraging GMM, GMR and Mahalanobis distance [137], that we are going to refer with the M1 tag.

3.4.1 Data

The dataset used for this study is the same described in Section 3.3.2 and the considered gestures are pictured in Figure 3.4. The two methods under evaluation share the moving horizon window mechanism. However, given the intrinsic difference between the two methods some differentiation are necessary, as presented in Figure 3.11. The SLOTH method uses a window of length $N_S = 40$ samples at 10 Hz (4 seconds) while, M1 uses a window of length $N_{M1} = 95$ samples at 40 Hz ($\simeq$ 2.4 seconds). This difference comes from the different requirements that the training data should satisfy for the two methods. High-frequency data do not improve SLOTH performances but increase the dataset size, increasing training time. M1 is particularly sensitive to the sequence alignment, therefore data included in the training set have been cherry-picked, precisely aligned and cut resulting in a shorter temporal window.

3.4.2 M1

This method relies on two modules, working, respectively, off-line and on-line. The off-line *model builder* module is devoted to the creation of probabilistic models of human motions, starting from a provided set of human examples, and it relies on Gaussian Mixture Modelling and Gaussian Mixture Regression. Each model is defined as an expected curve and associate

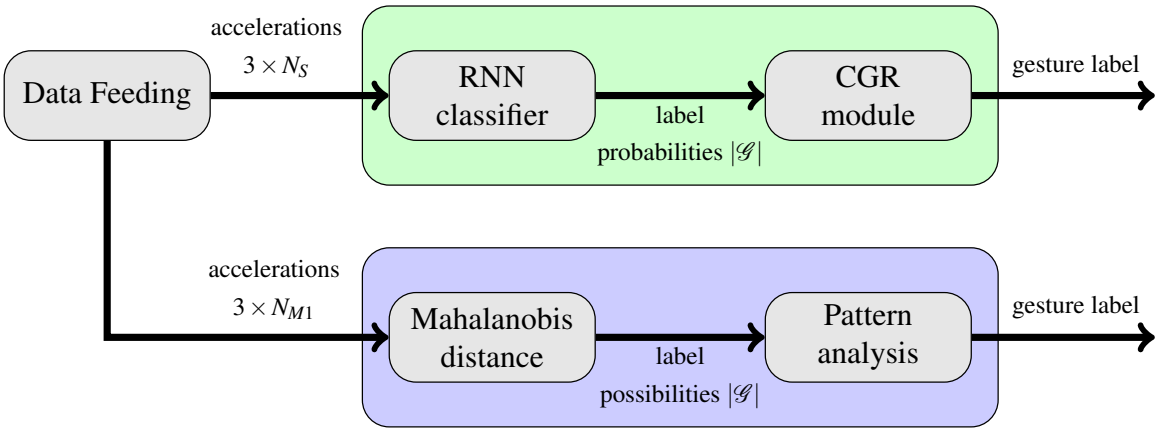


Figure 3.11 Evaluation architecture of the SLOTH method (top) and M1 method (bottom). The *Data Feeding* module feeds the two systems simultaneously but with different frequencies, to comply with the methods' requirements.

covariance. M1 separately models body and gravity accelerations, by extracting the gravity acceleration using a type II Chebyshev low-pass filter. The filtering process introduces a temporal delay that is handled by shifting back the signal losing the last d samples in the window. In order to avoid to miss a relevant part of the signal because of the filtering, if N_G is the maximum temporal length over all the gestures classes, the window length N_{M1} must be chosen as

$$N_{M1} = N_G + d. \tag{3.10}$$

The on-line *classifier* module labels run-time acceleration data by comparing them with the available models, and it relies on Mahalanobis distance [140] and a threshold mechanism to identify the model more likely corresponding to the run-time data, if any. More specifically, at each new sample the *classifier*, computes the possibility π_g (in the following, simply referred to as π) of each gesture class g to be related to the data within the moving horizon window. Whenever a gesture occurs, the possibility of the corresponding class rises, to reach a maximum (π_{max}) when the moving horizon window is perfectly aligned with the gesture (gesture end), and then decrease following a characteristic rise-fall pattern. The threshold mechanism triggers the gesture recognition when the rise-fall pattern is recognized: concretely, it first checks that the peak is high enough ($\pi_{max} > \eta$) and then waits for the fall portion of the pattern to start, to notify the recognition ($(\pi_{max} - \pi) > \gamma \pi_{max}$). Since the possibility reaches its maximum at the end of the gesture, the recognition can be triggered only after this instant, with an additional delay caused by the check on the fall pattern.

To train the models, *Dataset A* has been divided into a training set (11% of the sequences in the original dataset) and a validation set (89% of the sequences in the original dataset). The

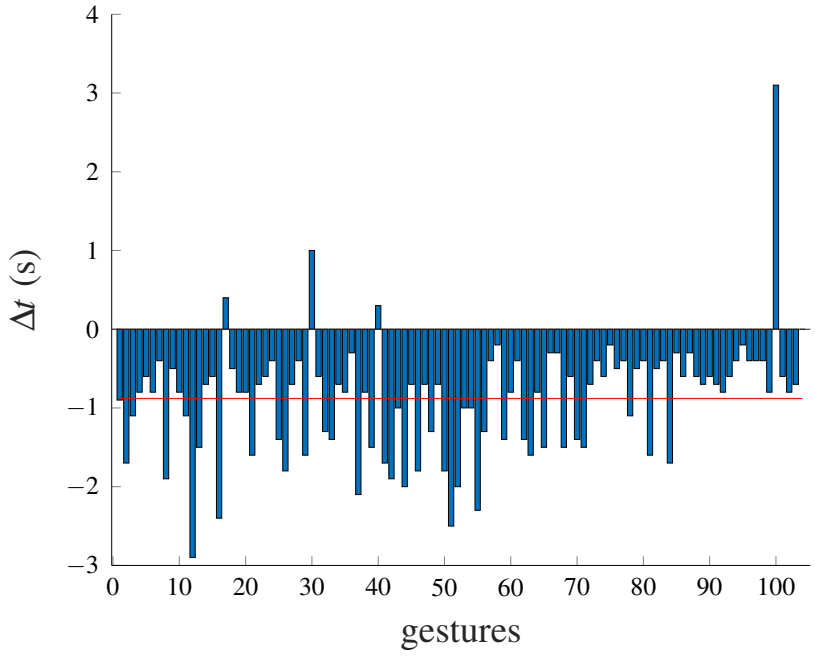


Figure 3.12 The time difference, computed as described in Eq. 3.11, between M1 and SLOTH using parameters presented in Eq. 3.12. Bars denote the time difference for the recognition of a single gesture, while the red line denotes the average difference over all gestures.

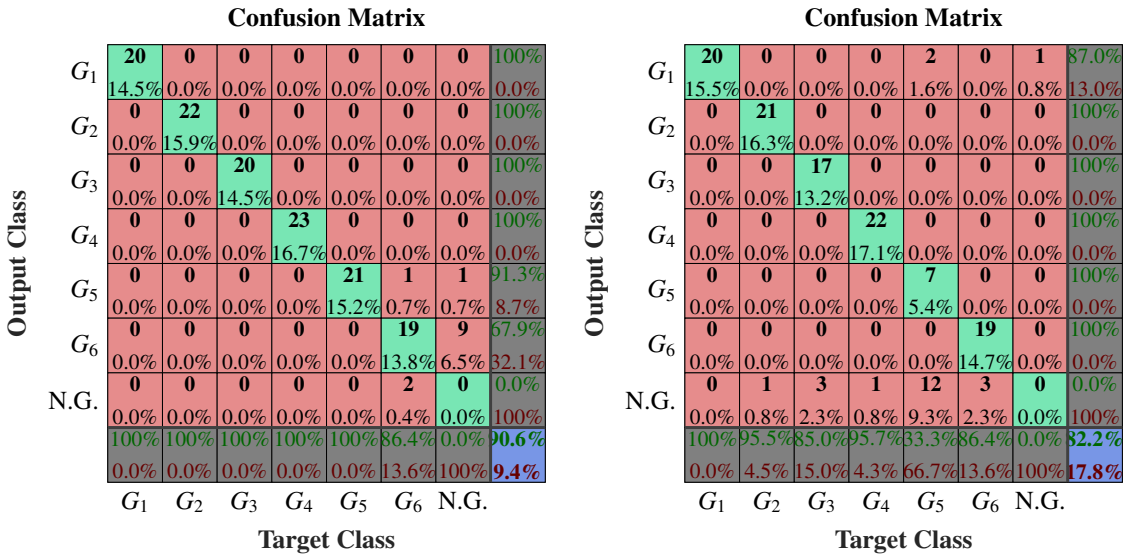
sequences to be included in the training set have been cherry-picked, to reduce the covariance of the generated models, and aligned. Furthermore, all the sequences have been extended at the end with d constant samples to prevent losing relevant data because of the filtering.

3.4.3 Experimental Evaluation

SLOTH and M1 implementation code is freely available on GitHub[1]. The *Data Feeding* module in Figure 3.11 loads *Dataset B* files, creates a downsampled version of the sequence at 10 Hz, for the SLOTH method, and at each step feeds the two methods with the proper window of data.

The metrics over which we have decided to perform the evaluation are recognition performances (F1 score [141]) and recognition speed. This two metrics have been chosen considering the observations we have done in Section 2.4. In fact, previous studies have addressed how recognition performances and recognition delays play a role in determining the system usability and user satisfaction.

[1]https://github.com/ACarfi/gesture_recognition



(a) SLOTH confusion matrix.

(b) M1 confusion matrix.

Figure 3.13 Confusion matrices of the two methods (with SLOTH using parameters values reported in Eq. 3.12). The bottom row reports the *recall* measures, the rightmost column reports the *precision* measures. The blue cell reports the overall accuracy.

As previously discussed, M1 requires the gesture end to perform the recognition: for this reason, the recognition time (t_{M1}) of M1 is used as a baseline for the comparison. Denoting with t_S the SLOTH recognition time, for the comparison we consider the parameter:

$$\Delta t = t_S - t_{M1}. \tag{3.11}$$

For each gesture instance successfully classified by the two methods the parameter defined in Eq. 3.11 is computed. At the same time, the results of the two methods are evaluated in terms of precision and recall.

The tests are carried out with two different sets of SLOTH parameters, selected from the one we have previously investigated:

$$\begin{aligned} \boldsymbol{C} &= [20, 20, 19, 19, 20, 20] \\ \boldsymbol{\eta} &= [0.598, 0.598, 0.582, 0.597, 0.56, 0.55] \end{aligned} \tag{3.12}$$

$$\begin{aligned} \boldsymbol{C} &= [3.9, 3.9, 3.85, 3.85, 3.9, 3.9] \\ \boldsymbol{\eta} &= [0.896, 0.896, 0.873, 0.895, 0.84, 0.825] \end{aligned} \tag{3.13}$$

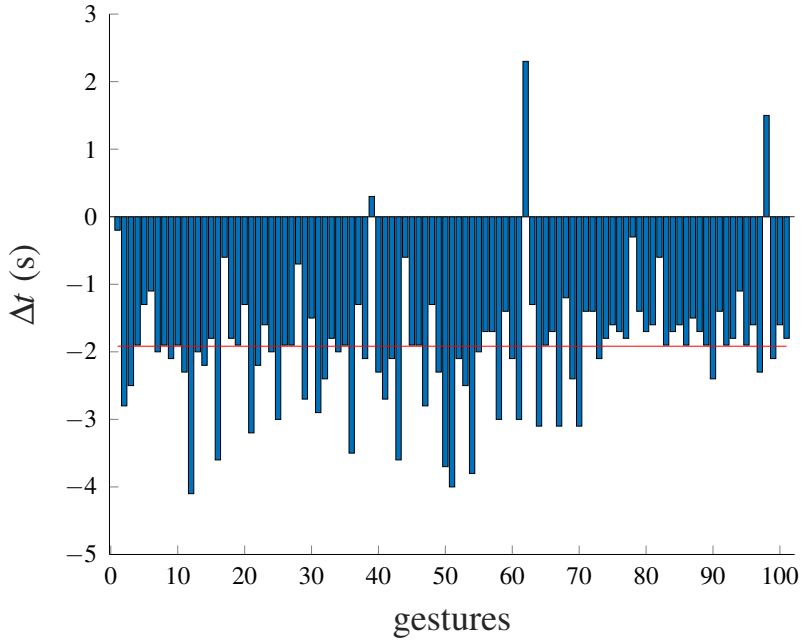


Figure 3.14 The time difference, computed as described in Eq. 3.11, between M1 and SLOTH using parameters presented in Eq. 3.13. Bars denote the time difference for the recognition of a single gesture, while the red line denotes the average difference over all gestures.

where the former configuration prioritizes the recognition performances, while the latter prioritizes the system reactivity achieving shorter recognition time. On the basis of the recognition performance over the validation set, the final models have been selected and the parameters of M1 have been set, in all experiments, as:

$$\begin{aligned} \eta &= 0.5, \\ \gamma &= 0.8. \end{aligned} \tag{3.14}$$

Figure 3.12 reports an analysis of the recognition times of SLOTH (using the parameters values reported in Eq. 3.12, which prioritize the recognition performance) and M1. Over the 128 gesture instances present in *Dataset B*, the number of gestures correctly recognized by both methods is 104. In Figure 3.12, each bar refers to Δt computed according to Eq. 3.11 for one specific gesture instance, while the red line represents the average difference computed over all 104 instances. As expected, the SLOTH method, that can recognize the occurrence of a gesture even before its end, achieves better recognition times with respect to M1 (on average, it is 0.9 s faster than M1, with a variance of 0.6 s^2), that, by design, has to wait until the gesture end to notify the occurrence of a gesture. Furthermore, as shown in the two

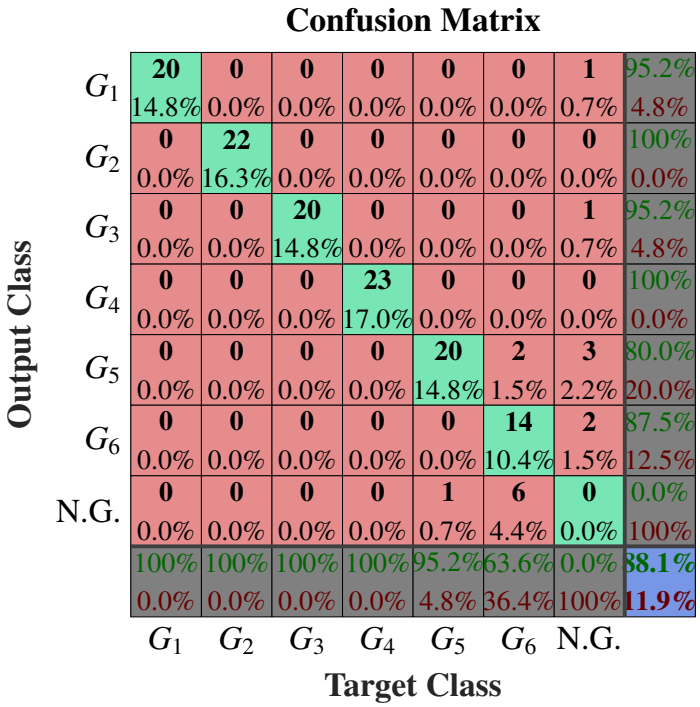


Figure 3.15 SLOTH confusion matrix using parameters values reported in Eq. 3.13. The bottom row reports the *recall* measures while the rightmost column reports the *precision* measures. The blue cell reports the overall accuracy.

confusion matrices in Figure 3.13, the SLOTH method has significantly higher recognition performances than M1, with an F1 score of 95.3% against an F1 score of 87.2% for M1.

Figure 3.14 reports an analysis of the recognition times of SLOTH (using the parameters values reported in Eq. 3.13, which prioritize recognition time) and M1. In this case, the number of gestures correctly recognized by both methods is 101, out of the 128 gesture instances present in *Dataset B*. As before, each bar of Figure 3.14 refers to Δt computed according to Eq. 3.11 for one specific gesture instance, while the red line represents the average difference computed over all 101 instances. As expected, with this set of parameters values, the SLOTH method is faster than with the first set of parameters values, and on average it takes 1.9 s less than M1 (with a variance of 0.9 s^2) to recognize a gesture. Figure 3.15 shows the confusion matrix of SLOTH, with the parameters values reported in Eq. 3.13: although, as expected, the recognition performance is reduced with respect to the first set (F1 score of 92.6%), it is still better than M1.

As the results show, the two methods achieve similar, good recognition performances, with the SLOTH method reaching high values both for precision and recall, while M1 prioritizes the precision. Moreover, SLOTH is significantly faster than M1, notifying the recognition of a gesture more than 1 s before M1. The main advantage of M1 over the

SLOTH method is represented by the modularity. Indeed, while adding or removing a gesture can be easily done with M1, the SLOTH method requires the retraining of the whole network, possibly compromising the performances of the system.

3.5 Interface

[2] Now that we have assessed the performance of SLOTH and compared them with another relevant approach proposed in the literature we have designed the interface with which the users is going to interact. As described in Section 3.1 the usage of a menu-based interface fixes the number of gestures required for the interaction while maintaining the possibility to add new functionalities.

This interface is meant for HRI in an industrial scenario. Therefore, we target lightweight manipulators capable of safely interact with the human operator. In our specific case, we used Baxter (see Figure 3.17) a dual-arm CoBot but the interface could be easily extendable to other platforms. Baxter is already equipped with the screen that we use to display the GUI. Therefore, the extension for another platform would require the pairing with a monitor if not already available. Baxter, as many other CoBot, already implements a functionality, through a software and hardware architecture, that lets the user guide the robot arms by mean of physical contact. This can be used in the framework of programming by demonstration (further details in the next chapter) to teach new task to the robot. Therefore, the main role of our gesture-based interface is to allow to activate such functionality to teach and play-back desired task. Notice that the interface is generic and whichever functionality can be integrated.

3.5.1 Working principle

In Section 2.4.1 we have described a gesture dictionary has a pair of command and their gestural expression. However, up to now we have considered and described only the gestural expression. To associate the gestural expressions we have used up to now to test SLOTH to their command we should, first of all, determine how the interface is going to look like and how the gesture will affect it. A screenshot of the main menu of the GUI is presented in Figure 3.16, we choose to distribute all the available options in a vertical configuration.

[2] The work described in this section has been carried out by Antonio Bongiovanni, Alessio De Luca, Luna Gava, Lucrezia Grassi, Alessandro Grattarola, Marta Lagomarsino, Marco Lapolla, Antonio Marino, Patrick Roncagliolo, Federico Tomat and Giulia Zaino and supervised by Alessandro Carfì for the course Software Architecture for Robotics, Robotics Engineering, University of Genoa.

Figure 3.16 A screenshot of the main menu of the GUI.

A red pre-selection indicator, in Figure 3.16 highlighting the Play option, is used to display the GUI status. To navigate such kind of menu we need at least three commands:

- up, moves the pre-selection indicator on the option on top of the current one;
- down, moves the pre-selection indicator on the option under the current one;
- select, triggers the behaviour associated with the preselected option: opens a new submenu or sends a set of predefined commands to the robot.

No going back command is needed since if an user enters in a submenu can come back selecting a back option. Referring to the gestures described in Figure 3.4 and used to evaluate SLOTH performances, our gesture command association follows: up (G3), down (G4) and select (G1). The rationale for this association is to keep up and down gesture the most short and simple as possible since are going to be the most used one and to preserve symmetry between gesture associated with opposite commands(up and down).

3.5.2 Implementation

Figure 3.17 presents the architecture structure for the overall gesture-based interface [3]. The sensing is performed using an LG G Watch R smartwatch and, for the data acquisition, has been developed an Android application directly communicating with the other architecture

[3]https://github.com/EmaroLab/gesture_based_interface

components through the Robot Operative System (ROS) framework. The previous implementation of SLOTH has been done in Matlab, therefore, for better integration with ROS, a new version of SLOTH has been developed using Python and Tensorflow [4].

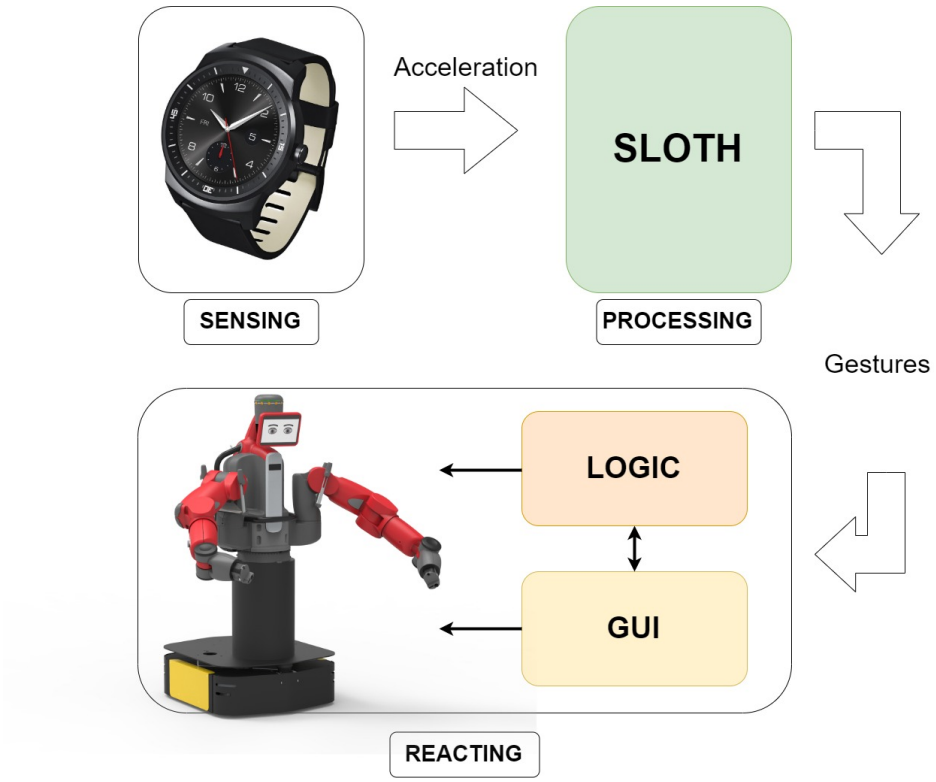


Figure 3.17 Software architecture for the gesture-based human-robot interface.

The logic behind the overall interface has been implemented using a finite state machine (FSM). This choice makes the system easy to be extended with new states. The state machine is implemented using the Python library SMACH and state changes are triggered by the recognition of a discrete gesture. The GUI interacts with the FSM to get the current state of the interface and therefore, change the visualized interface. The GUI is visualized on the Baxter screen. The activation of some FSM states do not trigger only a change in the GUI but are associated with commands addressed to the robot.

3.5.3 Subject study

To evaluate the first version of our gesture-based interface we have conducted a user study in which volunteers have been asked to perform a series of operations (teaching and reproducing

[4]https://github.com/ACarfi/SLOTH

some new tasks to the robot) using the gesture-based interface. To have a term of comparison the volunteers performed the experiments a second time using the same GUI deployed on a tablet so that they could interact using the touch screen, a technology with which persons are more familiar. Although a more in-depth experimentation and analysis of the results should be done to extract relevant conclusions, we would like here to discuss some hints we collected during the experiments.

It is evident that the selected gesture are inadequate for this application. The gesture dictionary could have been more intuitive if a different gestural representation would have been selected. In particular, gesture G4 is particularly difficult to be performed because of the human arm anatomy. Furthermore, all the gestures start and end with the person holding their arm still with their elbow at 90 degrees. This position is particularly unconformable and volunteers should maintain it for the whole duration of the experiment tiring the arm. We were already expecting these problems since, as we have stated at the beginning of this chapter, these gestures have not be designed taking in consideration the human factor but only to carry out a feasibility study.

Another problem that we encountered is a decay in the recognition performances of SLOTH, probably caused by a change in the data acquisition software. The data acquisition software previously used saves data on a file directly on the smartwatch and it had to be changed to continuously collect and stream the data. Although the data source remains the same, this software change can affect the consistency of the data frequency since some samples could be lost in the transmission process. Differences between the data used in the training process and the real data, especially due to differences in sensor and software [142], is a well know problem for data-driven approaches.

3.6 Follow Up

On the base of our previous study, we have planned and carried out some research activities aiming to improve our gesture based interaction system. In particular, we have focused on the sensing, the data processing and the dictionary design.

3.6.1 Sensing - Data Glove

At the beginning of this chapter, we have presented the reasons to use accelerometers for our application scenario. Since commercial solutions already integrate this kind of sensor in devices such as wristbands and smartwatches, we decided to opt for this solution forcing the

system focus on the arm. This choice allowed us to not focus on developing new sensing devices and to fast prototype our gesture recognition interface. Nevertheless, now we want to question this choice.

In our study, we are targeting industrial applications. In this kind of scenario the user is the worker, that should wear gloves. Therefore, it is possible to envision the sensorization of working gloves with IMUs since they are easy to embed in clothes. This would allow the system to include in the focus even hand and fingers enriching the set of gestures that the users can use to interact with the robot. For this reason, we have decided to design and develop a data glove solution for human gesture sensing.

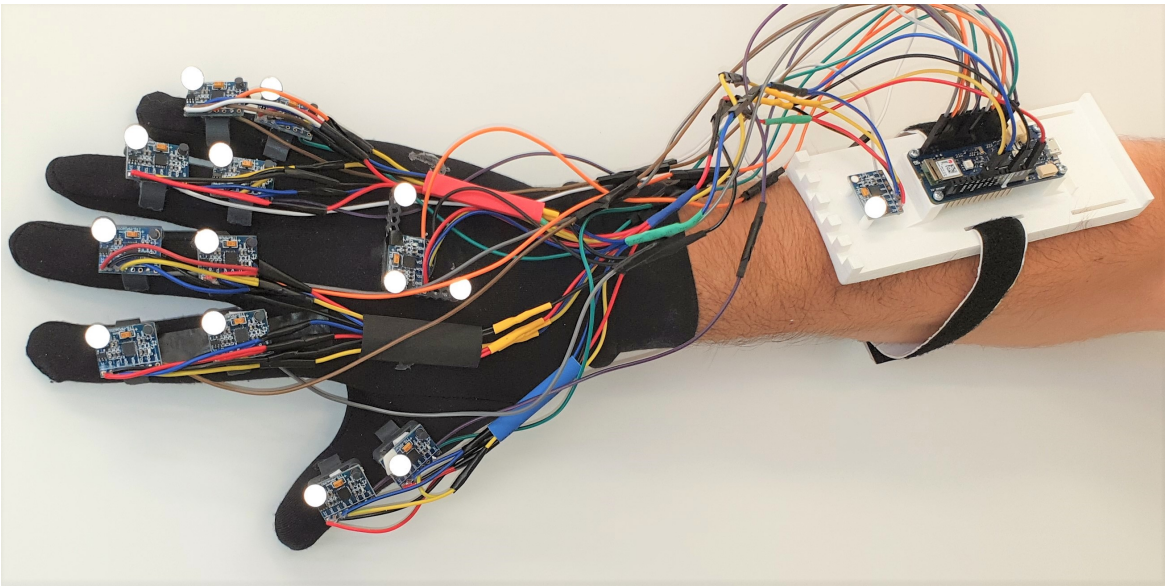


Figure 3.18 The data glove equipped with 12 IMUs, the MCU collecting the data and 14 reflective markers for the OptiTrack MoCap.

[5] The data glove is equipped with 12 6-axis IMUs (3-axis accelerometer and 3-axis gyroscope each) located as shown in Figure 3.18. Each finger is tracked by two IMUs, one on the intermediate phalanx and one on the proximal one. Furthermore, two IMUs are devoted to track the hand back and the wrist. The IMUs used for this prototype are the MPU-6050 from InvenSense providing a digital output through an I2C bus. The MCU used to control all the sensors is the MKR 1010, equipped with a WiFi module. To preserve the hand mobility and sensitivity a thin and flexible glove has been selected as support. To reduce their impact on the hand motion the IMUs are not directly glued on the glove but 3D printed mountings

[5]The work described in this section has been partially carried on by Federico Bernabei, Francesco Fallica, Francesco Giovinazzo, Giovanni Napoli, Maicol Polvere, Durgesh Salunkhe and Daniele Torrigino supervised by Alessandro Carfì for the first year group project.

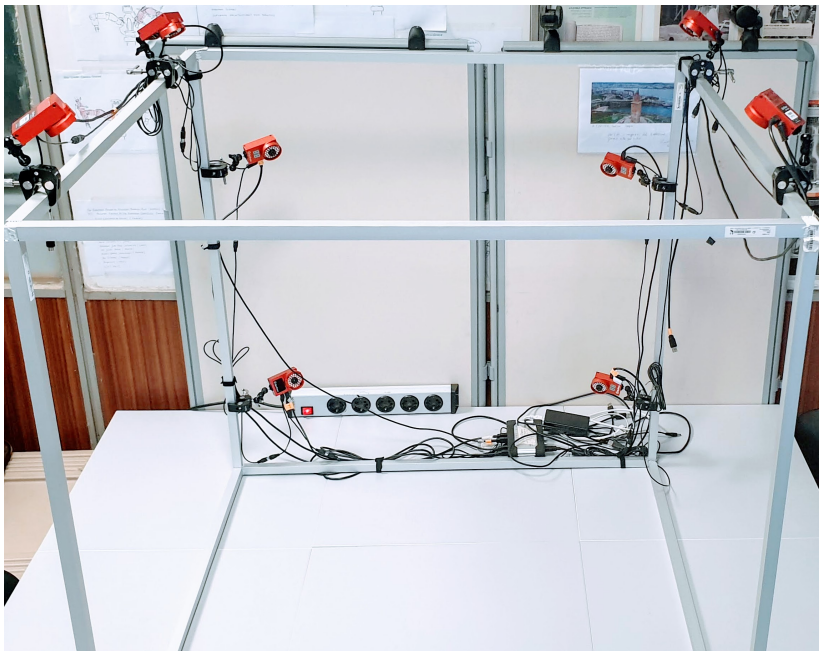


Figure 3.19 The structure supporting the OptiTrack MoCap.

have been used. The finger mountings consist of a half-ring structure while for the hand back an L-shaped support has been used. The wrist IMU and the MCU are mounted on a rectangular support fixed to the arm with a velcro strap. All the hand mountings have been printed using a flexible resin while for the wrist mount ABS material has been used. In Figure 3.18 can be noticed that each IMU is paired with one spherical reflective marker, three for the hand back. These markers are meant for other kinds of studies that we will describe in the last chapter and can be tracked by 7 Flex 3 cameras by OptiTrack[6] showed in Figure 3.19.

The software architecture is distributed among three devices. The MCU performs the start-up procedures to properly set the IMUs and samples the IMUs data. Because of the nature of the I2C bus, the sampling procedure should be performed sequentially. Each time a sample is acquired, the MCU sends it to a workstation through Wi-Fi using the UDP protocol. Each IMU sample contains the tri-axial linear acceleration and the angular velocity. Furthermore, the sensor microcontroller runs proprietary software to extract, from acceleration and velocity, the sensor orientation expressed as a quaternion.

3.6.2 Gesture Recognition - SPC

SLOTH achieves good results in our experiments relieving the close-world assumption that many works in the literature use to solve the gesture recognition problem. The main drawback

[6]www.optitrack.com

we have encountered with SLOTH is the lack of modularity. In fact, adding or removing a gesture would imply retraining completely the model losing any insurance on the system performances.

To solve this problem we introduce a new approach that we named Simultaneous Prediction and Classification (SPC). SPC takes advantage of a working principle introduced by Bailador et. al 2007 [143] where prediction error of continuous time recurrent neural networks (CTRNNs) predictors have been used to perform classification. The usage of a single predictor for each gesture guarantee the modularity of the overall system, although some further consideration on the performance changes is required.

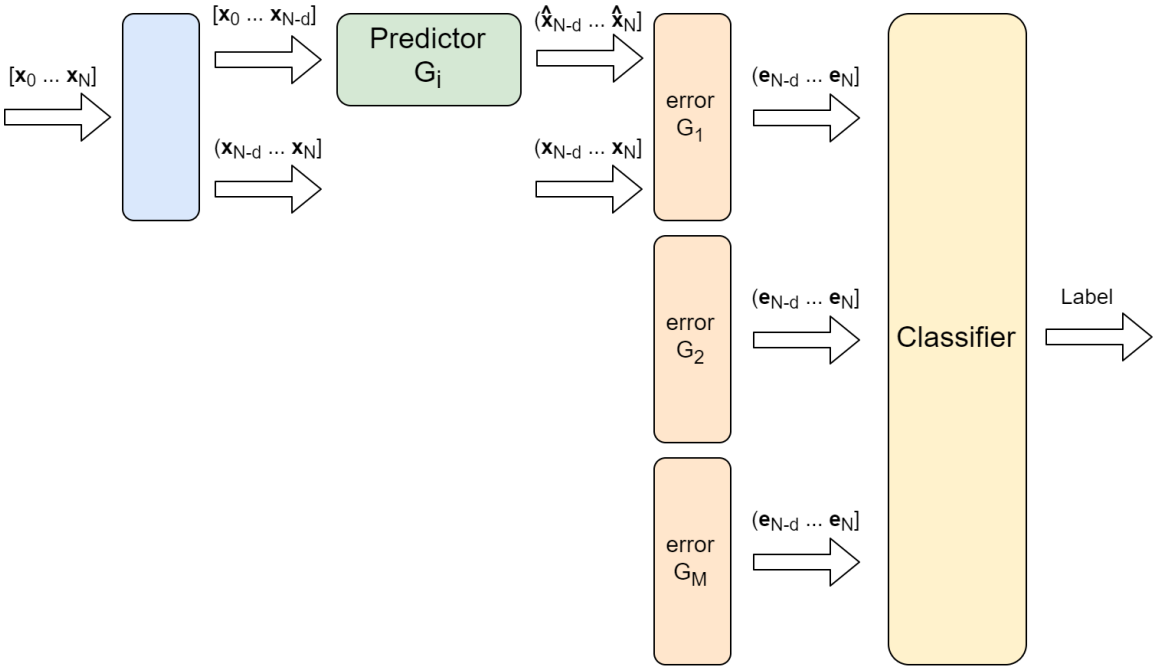


Figure 3.20 The working flow of the SPC method.

Working principle

Figure 3.20 graphically presents the working principle of SPC. For each gesture G_i is trained a Predictor G_i (Figure 3.20) that can predict up to a maximum of d samples. The first $N-d$ samples of the input vector, raw or preprocessed data, are fed to the Predictor G_i that returns the respective $d-$prediction values. The prediction values together with the real corresponding values are fed to the error block presented in Figure 3.20 that computes the prediction error. This process goes on in parallel for each of the M gestures the system has to

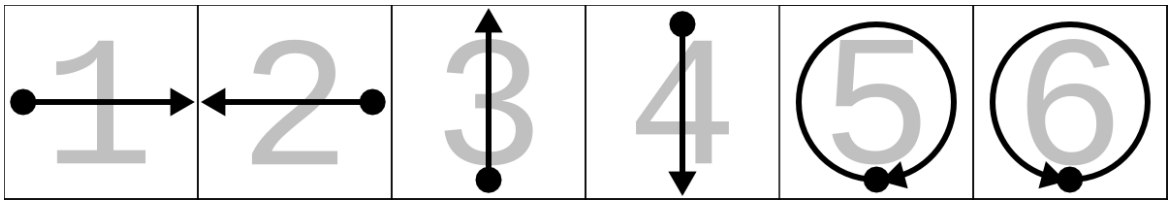


Figure 3.21 Gesture dictionary for SPC test.

recognize. Therefore, at each time instant are generate M prediction errors that are processed by a classifier to determine the label.

If the classifier is a probabilistic classifier, it is possible to use the detection module from SLOTH to perform the gesture detection. The method modularity is strictly related on how the classifier is implemented. In fact, if the classifier is implemented using a data-driven approach such as neural networks, it should probably be retrained if the gesture dictionary changes. Instead, by using a model-based approach the classifier could remain unvaried while changing the dictionary although the classification performances could vary since they are influenced by the predictors. Predictors are built to predict the last d steps of a specific gesture and SPC works on the assumption that a predictor has the lowest prediction error, among all predictors, when the fed data correspond to the gesture for which it has been built. However, it can happen that a predictor, built for G_i, achieves good prediction error even for G_j. To avoid this problem it is necessary to test each predictor with a bunch of different gestures, check the prediction error behaviour and change the predictor if its prediction performances are too good for gestures different by the one for which it has been designed. Therefore, it is possible to modify the gesture dictionary but it is important that every time all the predictors are tested to ensure everything is going to work properly.

Implementation

We have implemented SPC using RNN. Contrary to what we have seen in Section 3.3.1 we did not use a single RNN probabilistic classifier but we trained one RNN predictor for each gesture. The gestures and the data used for this experiments are not the ones described in Section 3.3.1 but a gesture dataset publicly available have been selected [144]. This dataset has been collected thanks to 8 different users performing 20 repetitions of 20 different gestures. Each sequence contains acceleration data from the 3-axis accelerometer of a Sony SmartWatch. Out of the 20 proposed gestures we have selected the 6 gestures presented in Figure 3.21 since they resemble the gestures used for the SLOTH implementation.

Since the dataset il smaller respect to the one presented in Section 3.3.2, we used 50% of the dataset for training and the remaining portion for testing. Six RNNs, one for each

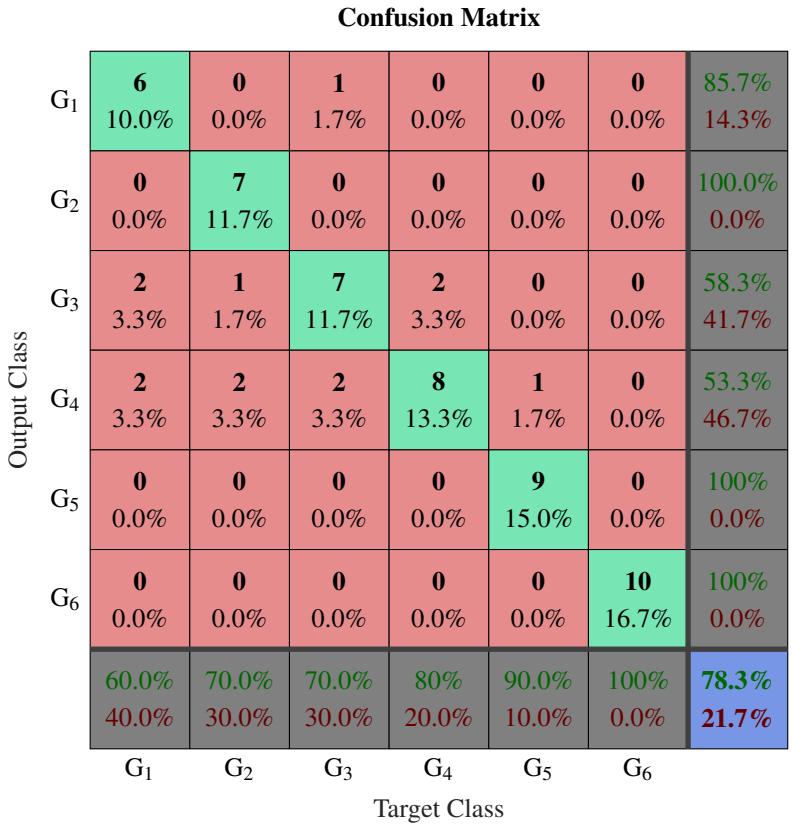


Figure 3.22 Confusion matrix for the SPC offline testing. The bottom row reports the *recall* measures while the rightmost column reports the *precision* measures. The blue cell reports the overall accuracy.

gesture, have been trained to predict the last value of the sequence, then the prediction error is computed using the euclidean distance. For each sequence in the test set the prediction error, for each of the 6 RNN, is computed and the network with the lower prediction error gives the label to the tested sequence.

As we can see observing the results of our preliminary tests presented in Figure 3.22, our SPC implementation does not achieve outstanding performances. Nevertheless, considering that SPC allows for a modular gesture recognition method, the results are promising. Further study should consider the usage of SPC for online recognition.

3.6.3 Gesture Dictionary

As we have notice in Section 3.5.3 the design and collection process of the gesture dictionary previously used to evaluate SLOTH and the gesture-based interface has some drawbacks. The low intuitiveness of gestures, in relation with the application scenario, high levels of

physical stress, associated to the gesture execution, and the usage of a software for the dataset collection unsuitable for the on-line usage that leads to differences between training data and the one collected on-line. Therefore, we have designed a new set of gestures and an experimental protocol for the data collection.

Gesture Design

The design flaws related to the gestures design involved both gesture intuitiveness and high physical stress. Gestures to be intuitive should be related to the application. We want to design a gesture set that could be used with whichever kind of menu-based interface. With this intent we selected six gesture[7]:

- up (Figure 3.23);
- down (Figure 3.24);
- left (Figure 3.25);
- right (Figure 3.26);
- push (Figure 3.27);
- pull (Figure 3.28).

These gestures have been chosen to be easily related with the corresponding action: up gesture would move the red pre-selection on the option on top of the current position; down gesture would move the cursor on the option under the current one; push gesture would trigger the behaviour associated with the pre-selected option; pull gesture would exit the current process, or menu, and call the previous menu.

All the gesture-action association have been thought to be straightforward and we included the left and right gesture to be able, if needed, to change the menu from a vertical organization to an horizontal one. Furthermore, we aim to reduce the physical stress associated to the gesture execution by imposing as start and end pose a relaxed configuration with the arm laying down on the user side and the hand palm facing the leg. The gesture figures present the gestures division in preparation, stroke and retraction. Notice that during the stroke phase of each gesture the palm is facing toward the movement direction.

[7]Credit for the drawings to Marina Lemucchi.

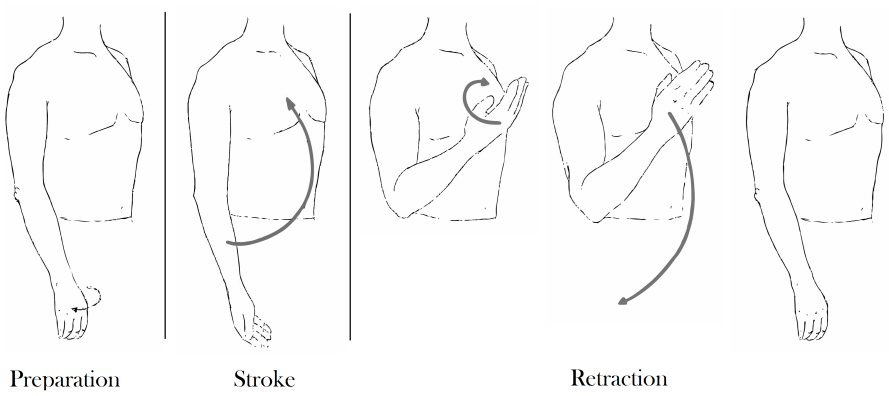


Figure 3.23 Up gesture.

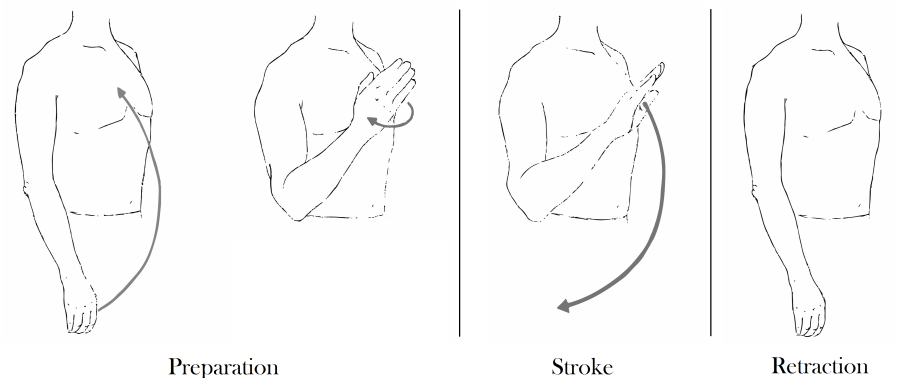


Figure 3.24 Down gesture.

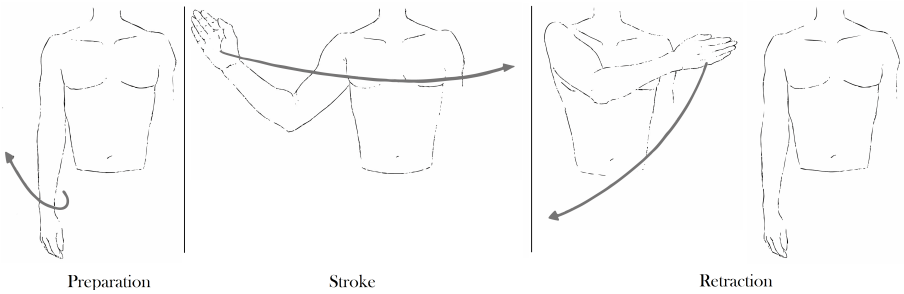


Figure 3.25 Left gesture.

This gesture dictionary has been designed to be intuitive, for the selected application scenario, and to prevent physical stress. Nevertheless, before moving on to collecting this dataset three experimental evaluations should be conducted.

First of all, we have to determine if the graphical representation we have designed for each gesture is effective in describing the gesture. In Section 2.4.1 we have described why being

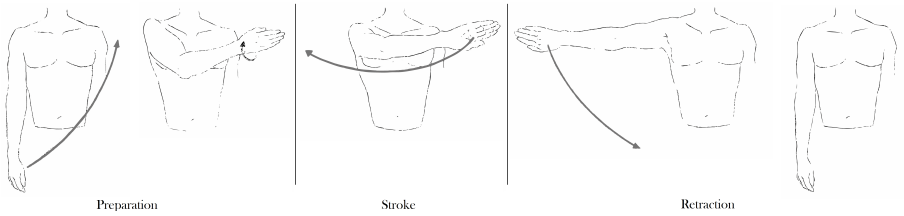


Figure 3.26 Right gesture.

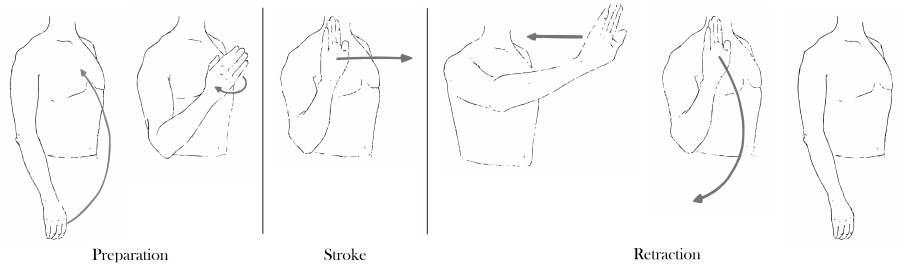


Figure 3.27 Push gesture.

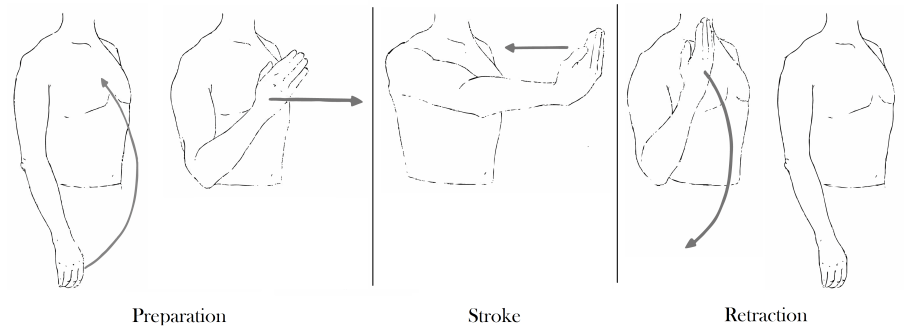


Figure 3.28 Pull gesture.

able to precisely convey the gesture description is fundamental both for reproducibility of the work and to provide an appropriate user manual. Therefore, we should conduct qualitative experiments to asses if persons presented with the gestures drawings and a brief description are able to perform the gestures as we have intended.

Then the user physical stress and mental effort, associated to the proposed gestures, should be evaluated to determine that the new dictionary has been properly designed to take in consideration the human factor (Section 2.4). The evaluation can be conducted using quantitative measurements such as questionnaires answers and electromyography signals, often used to evaluate human fatigue.

Finally, we should determine whether the new gesture dictionary is intuitive for our specific application. This can be done by carrying out a Wizard of Oz experiment in which the volunteer normally interacts with the gesture-based interface but an experimenter is in charge of sensing and recognizing the gestures. This experiment can be conducted twice, once with the old and once with the new dictionary, to evaluate if the new design improves the system intuitiveness.

If the results of the proposed experiments are negative it would be necessary to correct the gestures design accordingly and repeat the experiment as long as a correct design is identified.

Data acquisition

Once the dictionary has been fixed and properly evaluated it is necessary to collect a new dataset. Has we have seen in Section 3.5.3 our dataset had some problems connected with the procedure used for the collection. Therefore, we have redesigned the experimental protocol.

The gestures are going to be perceived by multiple devices such as the data glove, multiple android smartwatches and iOS smartwatch. The increase of the number of devices used in the dataset collection aims to reduce the sensitivity of trained models to sensor and software changes. Furthermore, all the considered smartwatches should be worn in different position of the arm to account for different ways in which users can wear the devices. Finally, the software used for the dataset collection should be the same used by the final application. Therefore, the data are not going to be saved locally on the device but will be streamed continuously to a PC that is going to save them in a file.

To facilitate the dataset collection a GUI has been developed [8]. After a setup face in which the GUI asks the volunteer to insert personal information such as age, height and dominant hand, the software verifies that all the sensing devices are connected. If all the communication with all the sensing devices works properly the collection process starts. The GUI shows the image of the gesture to perform (see Figure 3.29) to the volunteer and waits n seconds before showing a new gesture (remaining time is displayed by a progress bar), the procedure is repeated to collect p repetition of each gesture in the dictionary. The n parameter can be used to influence the speed that volunteers have in performing a gesture, different n value can be used to have a more varied dataset. The order in which the gestures are performed is random and different for each volunteer. The data are saved as a continuous stream. Between each gesture execution, the user is asked to stand still in the start/end posture, therefore using the timestamps in which new gesture drawings are shown to the user,

[8] https://github.com/ACarfi/gestures_dataset_collection

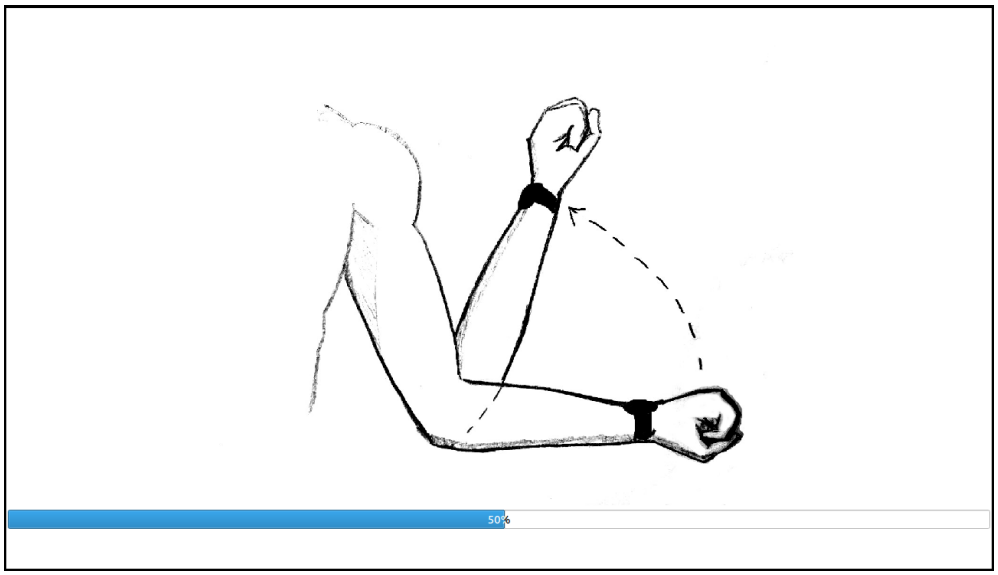


Figure 3.29 A screenshot of the GUI developed for data collection. The blue progress bar shows the remaining time before the next gesture is displayed.

the data can be tagged and segmented automatically. Sensing data are synchronized between themselves and with the data from an RGB camera used for debugging purposes.

Chapter 4

Continuous Gestures

4.1 Preliminary Considerations

In this Chapter, we focus on a specialization of the problem (Continuous GI-UI), by considering the usage of continuous gesture in *kinaesthetic teaching* (KT). KT (Figure 4.1) is a teaching technique, well-suited for robot manipulators, which assumes a human operator to physically move the robot, by means of continuous gestures, in the execution of the target task, while the robot records the movements of its joints. In KT, the continuous human gestures are mapped to the continuous robot joints state to allow the robot to follow the human motion. KT is used in the *Programming by Demonstration* (PbD) framework, referred in Section 3.5 while listing the available functionalities of our gesture-based interface.

In Chapter 1 we envisioned the requirements associated with smart factories. As a consequence, the Industry 4.0 paradigm requires the availability of techniques to enable a fast and easy-to-attain robot task reconfiguration. PbD has been envisioned to address such requirements advocating methods to teach robots new tasks *intuitively* [145]. PbD assumes two programming phases, namely *teaching*, where one or different realizations of the target task are shown, and *learning*, in which examples are generalized in order to synthesize a resulting robot's behaviour. PbD has the competitive advantage of not requiring any robot-specific competence related for reconfiguration issues and task teaching [7]. In this paradigm, human operators tend to become specialized labourers whose knowledge mainly derives from hands-on experience gained by interacting with the robot during the teaching process. The lack of awareness of how robots work, their limitations, as well as the inherent differences between their and human motions, can lead to an overall low-quality *teaching* process when KT is adopted [146].

Depending on the adopted *learning* technique, an inaccurate *teaching* can have different effects on the final result. In particular:

- *pure playback* exactly replicates the taught example; this is the case in which a bad training has the worst consequences, since execution time is equal to teaching time, and therefore any inefficiency in the teaching phase is replicated during execution;
- *waypoints playback* optimizes execution time but requires a longer teaching time (e.g., an operator must stop at each key waypoint) and it does not constrain motion between pairwise waypoints, which means that their suboptimal selection can lead to inefficient motions;
- *generalization* over different examples reduces the influence of a single inaccurate example on execution time but increases the required teaching time.

The teaching quality is influenced both by operator's skills and by robot usage intuitiveness. The evaluation of teaching quality can provide robot designers with useful hints to improve the cooperation process between human operators and robots and consequently increase the acceptance of robot in shop-floors by humans.

The content of the following section are extracted from Carfì et al. 2019 [28].

4.2 Background and Definitions

4.2.1 Background

Programming by Demonstration [145, 7], when applied to Robotics, allows for the implementation of robot behaviours while avoiding the traditional programming workflow, which typically includes a formal definition of the end-effector's trajectory, its representation in joint space using an inverted Jacobian and its use as a set of reference poses for a closed-loop control system. PbD can be critical as far as human-robot interaction is concerned, and its naive use can lead to suboptimal or even inaccurate robot trajectories. In fact, solutions for trajectory corrections using tactile sensors [147] or graphical user interfaces [148] have been proposed in the literature. However, such companies as Universal Robots and Rethink Robotics, just to name a few, commercialize manipulators for collaborative work in industrial settings, which take advantage of PbD, such as UR3[1] and Baxter[2], but do not implement any mechanism for trajectory correction.

[1] www.universal-robots.com/products/ur3-robot
[2] www.rethinkrobotics.com/baxter

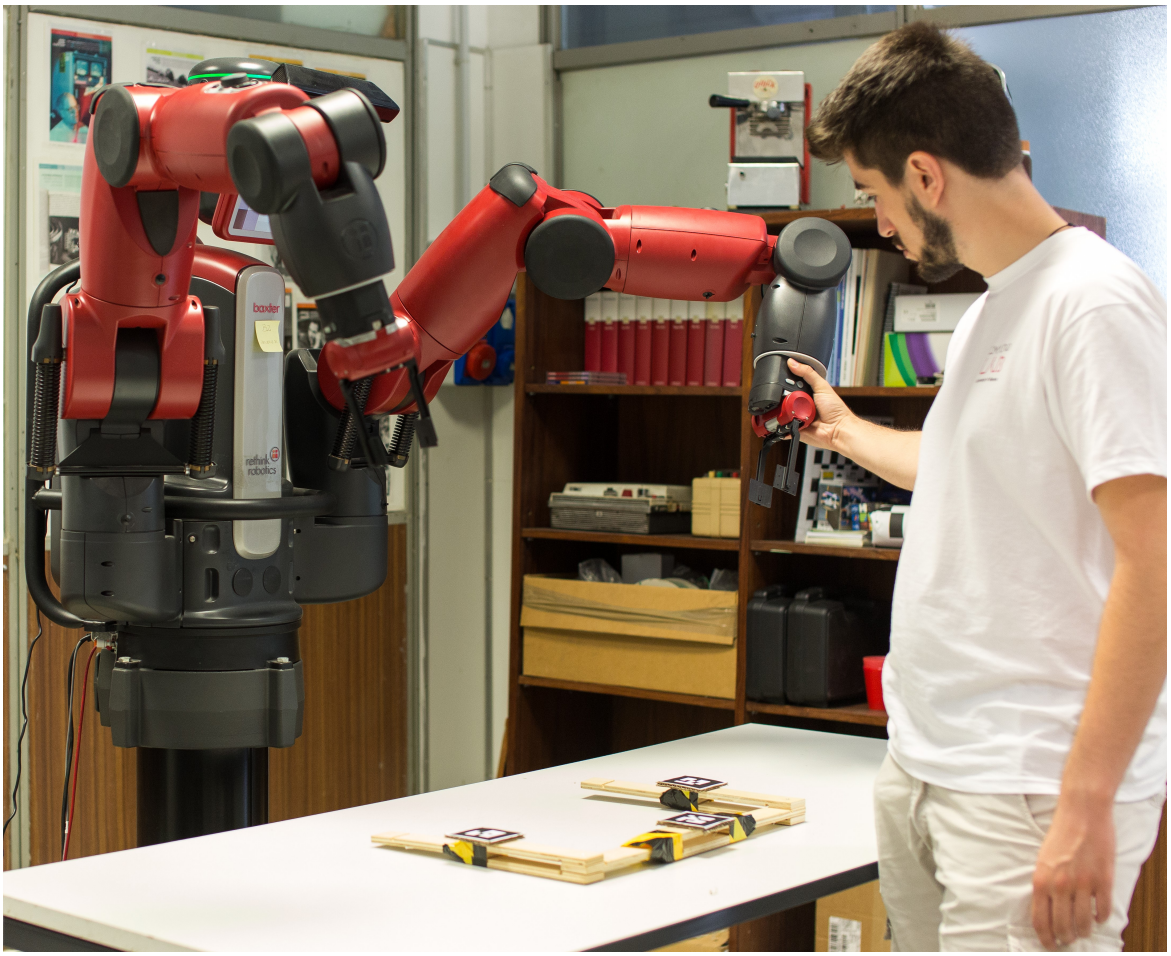


Figure 4.1 A human operator *teaching* a Baxter robot how to grasp an articulated object using Kinaesthetic Teaching.

As anticipated in the Introduction, the workflow associated with PbD is divided into two phases, namely *teaching* and *learning*.

In the teaching phase, an operator provides the robot with examples of the task. To this aim, different approaches can be adopted, from teach pendants to data gloves [149], as well as vision systems [150], or haptic devices [151], although PbD is usually implemented via *kinaesthetic teaching* [152][153][154][155][156]. In order to use kinaesthetic teaching, a sequence of operations must be carried out:

1. notifying when the teaching procedure starts through buttons on the robot or via a different input device;

2. manually operating the robot through a series of waypoints or following a desired path for the end-effector and, when a specific waypoint must be recorded, pushing proper buttons;

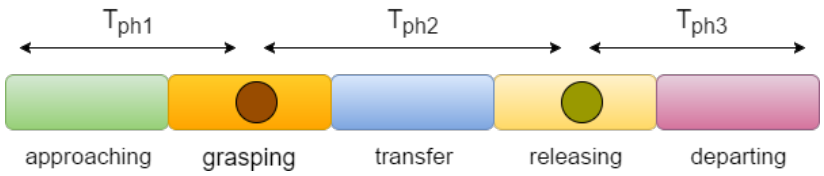


Figure 4.2 A visual representation of two classifications of a pick and place task, respectively separating the task into three temporal slots (top), and in five volumes of interest (bottom).

3. manually activating such actions for the end-effector as opening or closing the gripper;

4. notifying the robot when the teaching procedure ends.

Furthermore, when a dual-arm manipulator is used, an operator must coordinate the teaching process between the two arms, or perform the two teaching procedures separately and sequentially.

During the learning phase, the robot is expected to generalize the provided examples to obtain the final behaviour. To this aim, different approaches have been proposed, such as Neural Networks [157], Hidden Markov Models [158] or Fuzzy Logic [159]. Furthermore, the learning procedure is meant to filter errors, uncertainties and suboptimalities present in the demonstration because of the operator, e.g., wrong or inaccurate teaching motions. Surprisingly enough, there is scarce literature evidence on solutions aiming at *preventing* such disturbances introduced in the teaching phase, or even on systematically assessing their nature and impact on the final robot behaviour. Relying on advanced learning techniques in industrial scenarios can be troublesome due to their unpredictability. For these reasons, KT in industrial settings is usually based on recording way-points and no countermeasure is in place to mitigate the effects of inaccurate teaching.

4.2.2 Definitions

A pick and place task

Typical tasks in which industrial robots are employed consist of packaging, assembly, or loading and unloading of objects to and from conveyor belts. These tasks can be modelled as composite actions that require to reposition an object between two locations as a sequence of picking and placing actions. Since pick and place actions can be considered as archetype operations for a variety of other tasks, here we put forth a few general definitions for pick and place tasks, which will be used throughout the paper.

Pick and place is a composite action in which an agent, in our case a robot, moves an object from a start pose to an end pose, following a trajectory. Each pose has linear

($p \in \mathbb{R}^3$) and rotational components, therefore the linear component of the trajectory is $\tau(t) = \{p_1, \ldots, p_{|\tau|}\}$ such that $p_1 = p_s$ and $p_{|\tau|} = p_e$. From now on we refer only to the linear components of trajectory and poses. The trajectory can be divided into different chunks, depending on whether one considers a *temporal* or *spatial* classification (Figure 4.2). These two classifications are aimed at capturing different aspects of the pick and place task. In a temporal classification, it is possible to identify key time instants such as *grasp* t_γ, i.e., when the gripper on the end-effector is closed around the object to pick in point p_γ, and *release* t_r, i.e., when the end-effector is opened in p_r. Figure 4.2 shows grasp and release time instants using a red and a green circle, respectively. The implicit assumption that grasp and release actions are instantaneous induces three *well-defined* temporal intervals delimited by t_s (start time), t_γ, t_r and t_e (end time), namely $T_{ph1} = \{t_s, \ldots, t_\gamma - 1\}$, $T_{ph2} = \{t_\gamma + 1, \ldots, t_r - 1\}$ and $T_{ph3} = \{t_r + 1, \ldots, t_e\}$. In a spatial classification, the trajectory τ is divided into five *qualitative* spatial intervals or portions of trajectory depending on their semantics (shown in Figure 4.2 with different colours):

- *Approaching* T_a includes all points corresponding to the phase in where the robot reduces the distance between its end-effector and the object to pick, moving from p_s to p_γ;
- *Grasping* T_γ groups all points in which the robot is trying to identify a suitable grasping pose, roughly clustered around p_γ;
- *Transfer* T_t clusters all points corresponding to the actual relocation of the object from p_γ to p_r;
- *Releasing* T_r includes all points in which the robot aligns the object with the desired release pose p_r and releases it;
- *Departing* T_d groups all points corresponding to the phase in which the robot leaves p_r to reach the end position p_e.

Neighbourhood

In order to characterize the notion of spatial distribution, we extend the mathematical concept of *neighbourhood* of a point [160] as follow. Given a point $p \in \mathbb{R}^3$, we define its neighbourhood N_p as a set of points $c \in \mathbb{R}^3$ such that the Euclidean distance δ_{cp} between c and p is lower than a threshold Δ, i.e., $\delta_{pc} \leq \Delta$. Then, for a given trajectory τ, and for each

point $p \in \tau$, the cardinality $|N_p|$ is used to define a local spatial distribution, or density $d_{\tau,p}$ as

$$d_{\tau,p} = \frac{|N_p|}{\Delta^3}. \tag{4.1}$$

4.3 Rationale and hypotheses

The requirement of frequent robot reconfiguration implies that such an operation is fast and easy-to-attain. As we discussed in the previous Section, PbD seems a reasonable solution, since it proves to be quite intuitive also for non-technically skilled operators [161]. PbD makes it possible to setup a robot for operators without any specific knowledge about how robots work, but with practical knowledge of traditional industrial equipment. This can lead to [problem P_1] suboptimal solutions as far as a trajectory τ is concerned, or the need of adopting more complex and robust learning approaches to overcome suboptimalities, which typically leads to a longer overall teaching time $t_{|\tau|}$. Since different operators have different working experience, [insight I_1] an experienced operator is likely to obtain better results at teaching than a less experienced one. Furthermore, operators with different skills and experience may spend a different time for the same teaching goal, which could [P_2] tempt less skilled operators to *speed up* the teaching procedure to the detriment of final results. Indeed, [I_2] the necessity for a fast reprogramming, lack of time, stress, inexperience or laziness are all factors that may cause a reduction of the teaching time, with possible negative consequences on the final trajectory.

Starting from these considerations, the following hypotheses are in order. [Hypothesis H_1] The time required to teach a robot a given task, comprehensive of operator's attempts and the number of iterations required by the selected learning approach, varies among operators. In particular, [$H_{1.1}$] teaching time varies with the operator, and [$H_{1.2}$] considering the different phases, the teaching time required for T_{ph1}, T_{ph2} and T_{ph3} (Figure 4.2) differs for different operators. These hypotheses, if integrated with the previous observations, lead us to state that [I_3] the variation in teaching time, both overall and for different phases, can be reduced if the operator is trained to that aim. Considering the robot's end-effector trajectory τ, as executed at teaching time, [H_2] the two spatial intervals related to *grasp* and *release* instants are characterized by a higher density with respect to nearby points in the trajectory. In particular, [$H_{2.1}$] T_γ and T_r greatly differ when different operators are involved, while [$H_{2.2}$] the differences in T_a, T_t and T_d for different operators are lower. As a consequence, it may be reasonable [I_4] to reduce the operator's influence during T_γ and T_r because of their criticality in the overall trajectory.

In order to validate what discussed above, we posit the following working hypotheses:

WH_1) exposing operators to PbD training footage or to human-human interactions, in which one human role-plays as a robot engaged in PbD, reduces the variance of teaching time;

WH_2) the need to manually control the opening and closing of a robot's gripper is a disturbing factor for the operator, which affects how grasping and releasing actions are taught;

WH_3) the need to identify an appropriate grasping pose for the object to pick while teaching the task has a negative effect on the resulting robot trajectories.

4.3.1 Preliminary study

The experiment designed to test H_1 and H_2 is composed of two distinct activities: *Human-Baxter PbD* and *Human-Human PbD*. In the former activity the volunteer uses PbD with a robot, while in the latter with another volunteer. The *Human-Human PbD* activity is meant as a pre-training for the *Human-Baxter PbD*. Therefore, to evaluate the possible influence of volunteers pre-training [I_3], volunteers have been divided into two homogeneous groups, as shown in Figure 4.3, that perform the two activities in opposite order (i.e., with one group serving the purpose of a control arm). Analysis of quantitative data recorded during the *Human-Baxter PbD* has been used to evaluate H_1 and H_2. In particular two metrics have been considered: temporal duration of pick and place phases (as defined in Sec. 4.2.2) and neighbourhood density about the grasping and releasing poses (as defined in Sec. 4.2.2). Each activity takes around 45 minutes for completion and the whole experiment has been carried out in a single day in our lab.

Volunteers

The experiment involved 25 unpaid volunteers among the students and teachers of a vocational educational and training school in Italy, aged between 15 and 60 with a median value of 16. Students have previous knowledge of industrial equipment, with a median experience of one year, although not collaborative robots, and may well be thought of as possible robot operators in the *factories of the future*.

Human-Baxter PbD experiments

The setup of the Human-Baxter PbD activity is shown in Figure 4.4. A Baxter dual-arm manipulator stands in front of a table where two location, namely A and B, are defined. In A,

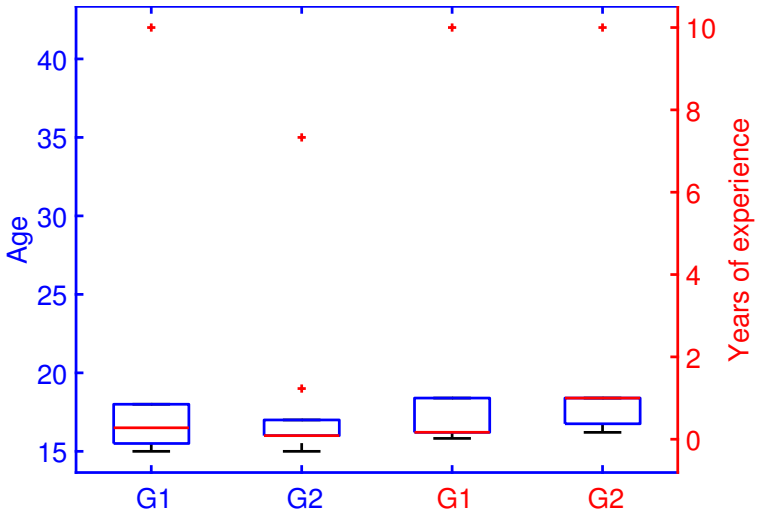


Figure 4.3 Volunteers statistics for each group related to age and years of experience with industrial equipment.

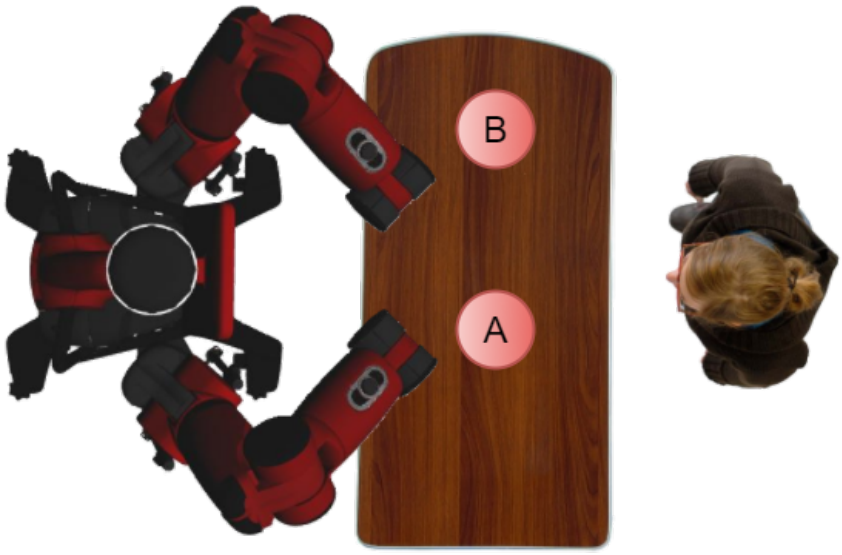


Figure 4.4 Experimental scenario of the activity *Human-Baxter PbD*: Baxter stands in front of a table where pick and place actions are performed.

a 0.5-litre plastic bottle filled with water for about one-tenth is located, whereas in B there is an open box with a 30 *cm* base and 12 *cm* height. The distance between A and B is 78 *cm* while the distance between Baxter and the table is around 60 *cm*.

Before the activity starts, an experimenter gives a practical demonstration about the use of KT to teach Baxter how to perform a pick and place task, i.e., teaching Baxter how to use its left arm to relocate the bottle from A to B, inside the open box, which is then performed by each volunteer. Each experiment starts with the Baxter in the *untucked* pose, while we did not constraint the final pose. The experiment loosely follows these steps:

i. a volunteer is located in front of Baxter, on the opposite side with respect to the table;

ii. the volunteer grasps Baxter's wrist to activate the zero gravity mode and starts the teaching procedure;

iii. the volunteer teaches Baxter how to relocate the bottle from A to B inside the open box using KT;

iv. when appropriate, the volunteer pushes the buttons located on the robot's wrist, to open and close its left gripper;

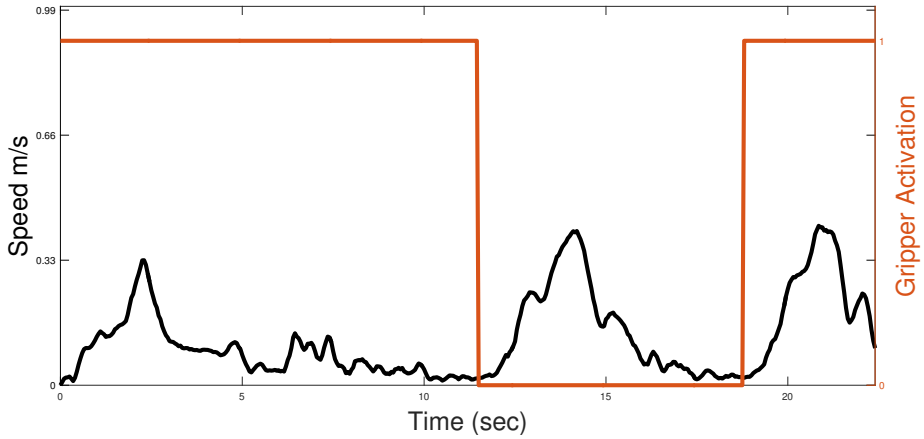
v. the task is executed: if the robot does not succeed in performing it, then the volunteer is asked to repeat the teaching procedure.

At each temporal instant, the trajectory $\tau(t)$ is recorded and expressed in joint space as $\sigma(t) = \{\phi_1, \ldots, \phi_{|\tau|}\}$, where for each point $p_i(t) \in \tau$ we define $\phi_i(t) = (q_1(t), \ldots, q_s(t))$. It is noteworthy that no constraints on teaching time have been set for this experiment.

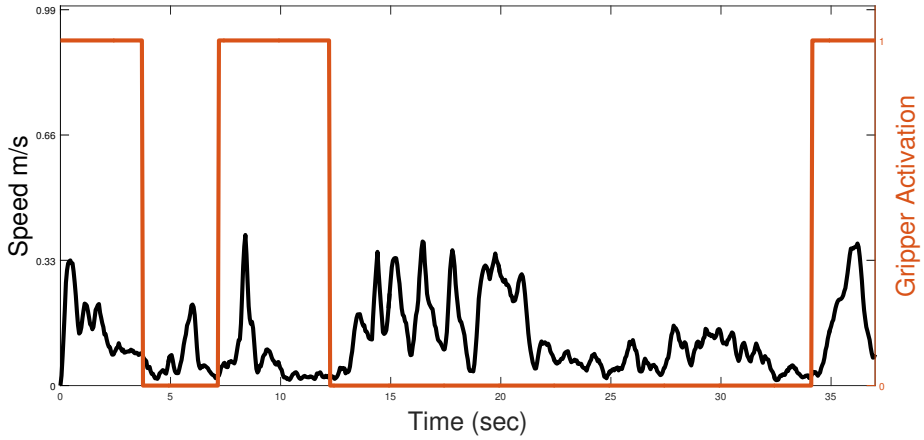
4.3.2 Human-Human PbD experiments

Before this activity begins, a 7-minutes video[3], intended to demonstrate how KT works on Baxter, is shown to the group of volunteers. Then, volunteers divided into pairs are asked to apply the same methodology to teach a predefined task to each other, while no verbal communication is allowed. Possible tasks include: closing/opening a jar lid, stacking six boxes forming a pyramid, ordering five bottles according to their weight, composing a square with four pens, picking and placing a box, folding a shirt, sinking a screw in a piece of wood. The experiment is repeated twice swapping the roles of *teacher* and *learner*. During all the experiments, teacher and learner are in front of each other with a table in between, as shown in Figure 4.13. Once the teaching procedure ends, the learner is asked to repeat the task.

[3] https://youtu.be/4FI7LwM3V38



(a) Volunteer in G1 highlighted in red in Figure 4.7

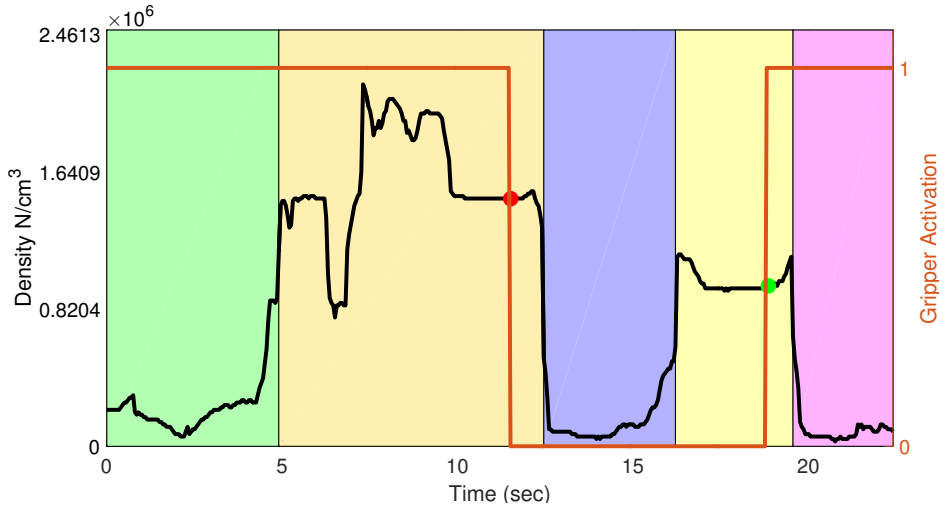


(b) Volunteer in G2 highlighted in red in Figure 4.7

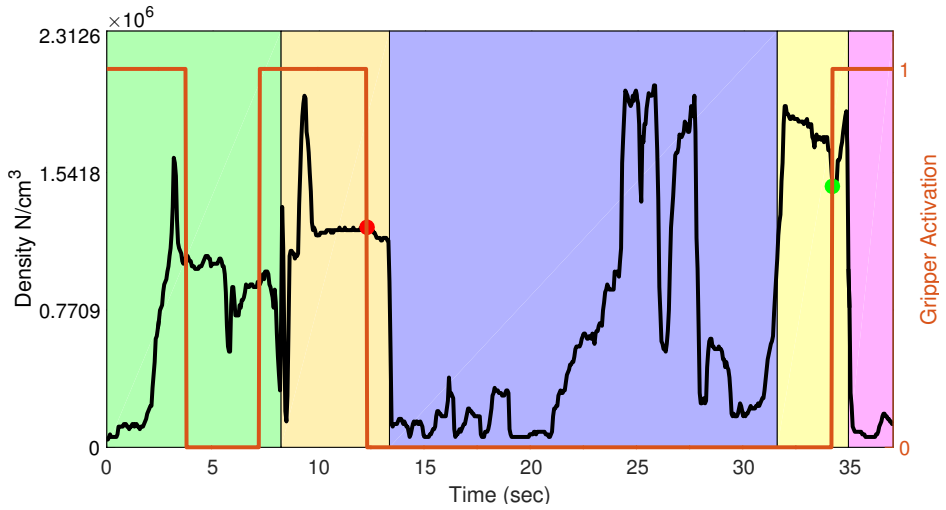
Figure 4.5 Black plots refers to the speed of the Baxter's end-effector, while red plots indicate the gripper status expressed as a Boolean value, where 0 stands for *closed* and 1 for *open*.

Analyses

We have carried out both a quantitative analysis related to the collected robot trajectories, recorded during *Human-Baxter PbD* experiments, and a qualitative assessment of video footage recorded during *Human-Human PbD* experiments. In this Section, we focus on the description of the quantitative analysis whose results are presented in Section 4.3.3, whereas Section 4.3.3 includes a discussion about what happens when two humans apply KT to each other.



(a) Volunteer in G1 highlighted in red in Figure 4.7



(b) Volunteer in G2 highlighted in red in Figure 4.7

Figure 4.6 The black plot represents, for each time instant associated to a trajectory τ, the number of *close* end-effector locations, i.e., the density distribution of the trajectory, whereas the red plot is the gripper status expressed as a Boolean value, where 0 stands for closed and 1 for open. Background colours refers to the pick and place *spatial classification* described in Figure 4.2.

Data collection and processing: All trajectories have been sampled at 100 Hz and are characterized by a variable time duration. We have downsampled the data at 20 Hz since it is enough to study the most common human activities. [162].

To collect data and convert them into joint space, we have used the Baxter PyKDL [4] library, we have analysed data using MATLAB R2015b. The trajectories have been processed in order to filter out initial and final volunteers lags in starting the demonstration and the experimenter lag in stopping the recording.

We are interested in three distinct analyses: the duration of the teaching process, the duration of task phases, and the spatial distribution of end-effector trajectory points. Assessing the duration of the teaching process allows us to investigate the variation introduced by different operators ($[H_{1.1}]$ in Section 4.3), and to understand whether such a variation can be reduced by a training session $[I_3]$. The duration of task phases is needed to observe how points in τ are distributed among T_{ph1}, T_{ph2} and T_{ph3} $[H_{1.2}]$. Finally, the analysis of density distribution is expected to assess whether T_γ and T_r, with respect to the other temporal intervals, are characterized by a higher density $[H_2]$.

Duration of the teaching process: We have computed the time $\Delta t = t_e - t_s$, required by each volunteer to carry out the teaching process.

Duration of task phases: This part of the analysis is based on the temporal classification, introduced in Section 4.2.2, which identifies three phases, namely T_{ph1}, T_{ph2} and T_{ph3}, separated by key time instants t_γ and t_r. The temporal classification is based on an empirical finding, determined during the present analysis, related to the speed at which the robot's end-effector is moved. Figure 4.5 displays two interesting cases. The *speed* (S) of the Baxter's end-effector is shown in black, while the gripper status (0 *closed* and 1 *open*) is in red. The *speed* is computed as:

$$S = \left\| \frac{\tau(t_i) - \tau(t_{i-1})}{dt} \right\| \tag{4.2}$$

for each point p_{i-1} and p_i in the trajectory, filtered using a moving-average filter. In particular, on the top, a recurring bell-shaped behaviour corresponding to the three phases outlined above can be observed, along with two low-speed parts of the trajectory roughly coinciding with the gripper opening and closing. It is noteworthy that this is not a peculiarity of a single experiment, but a feature characterizing the behaviour of the majority of volunteers. Figure

[4] http://sdk.rethinkrobotics.com/wiki/Baxter_PyKDL

4.5 on the bottom represents a case in which this does not happen. There is uncertainty in the gripper activation, as well as in the speed, but we can still observe that the speed increases and decreases with a bell-shaped pattern. This behaviour is probably due to the volunteer stopping or slowing down near locations *A* and *B* to push the buttons on the Baxter's wrist for gripper opening and closing.

Spatial distribution of end-effector trajectory points: With this analysis, we are interested in determining the differences in spatial distribution of end-effector's locations during T_γ and T_r with respect to the other phases. To this aim, we refer to the concept of *neighbourhood* introduced in Section 4.2.2.

We empirically fix Δ as the minimum number, considering all the experiments, that allows each point p in the trajectory to have at least one point in N_p, and we compute the density $d_{\tau,p}$ for each point in τ, for all the experiments. Δ is maintained constant for all groups.

The density $d_{\tau,p}$ of each experiment is used in order to divide τ in five intervals according to the spatial classification introduced in Section 4.2.2. The trend presented in Figure 4.6, whereby points p_γ and p_r are in high density intervals, is common to all the experiments. Particularly the normal density trend consists of two high density plateau as in Figure 4.6a. The two high density regions divide τ in five intervals that can be associated to the intervals defined in the spatial classification (Section 4.2.2): *Approaching* (T_a), *Grasping* (T_γ), *Transfer*(T_t), *Releasing* (T_r) and *Departing* (T_d). The *Grasping* interval starts at $t_{\gamma s}$ and ends at $t_{\gamma e}$ where $t_{\gamma s} < t_\gamma < t_{\gamma e}$, we define $t_{\gamma s}$ and them $t_{\gamma e}$ as:

$$t_{\gamma s} = \min_t ||t - t_\gamma|| \quad \textit{such that:} \begin{cases} t_{\gamma s} < t_\gamma \\ d_{\tau,p_{\gamma s}} \leq K \cdot d_{\tau,p_\gamma} \end{cases} \tag{4.3}$$

$$t_{\gamma e} = \min_t ||t - t_\gamma|| \quad \textit{such that:} \begin{cases} t_{\gamma e} > t_\gamma \\ d_{\tau,p_{\gamma e}} \leq K \cdot d_{\tau,p_\gamma} \end{cases} \tag{4.4}$$

where K is set arbitrarily to 0.8. The constraints in 4.3 and 4.4 define the start and end times of *Grasping* as the temporal instants closest to t_γ for which the density decreases to $0.8 \cdot d_{\tau,p_\gamma}$, respectively before and after t_γ. Since the choice of the density threshold is empirical we conducted a study on the distances between points within T_γ and p_γ to ensure that:

$$||p - p_\gamma|| < \varepsilon \qquad \forall p \in T_\gamma \tag{4.5}$$

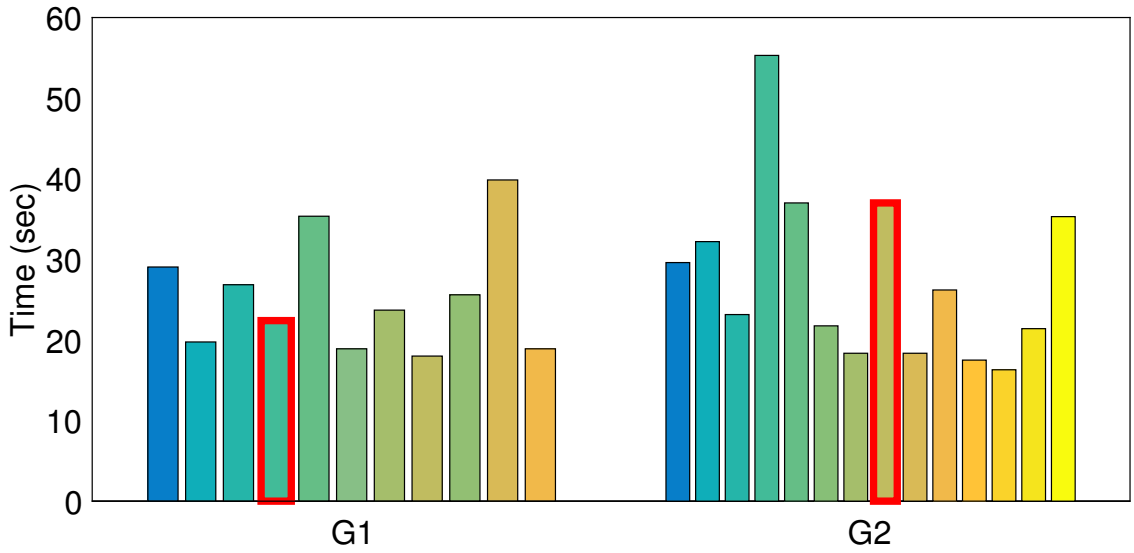


Figure 4.7 The time duration of the teaching procedure in the *Human-Baxter PbD* activities, for each volunteer, in groups.

The same approach is used to identify the *Releasing* location inside τ. Once the trajectories recorded for each experiment has been divided into the five intervals an analysis on the $d_{\tau,p}$ among them have been carried on.

4.3.3 Quantitative results

Duration of the teaching process: Figure 4.7 presents the time duration of the teaching procedure for all individuals, divided into the two groups. When considering the time taken by each individual, we observe that it is characterized by a high variance. In particular, G1 has a variance of about 50.62 s^2, whereas G2 has an even greater variance of 117.16 s^2. As expected, these results seem to confirm $H_{1.1}$, i.e., teaching time varies with operator, but the empirical cumulative distribution study presented in Figure 4.8 gives no evidence of a reduction of the teaching time due to prior knowledge related to the process or training, as argued in I_3 and posited in WH_1. In fact, the mean value for the two groups is almost the same (23.7 *sec* for G1 and 24.68 *sec* for G2), and even if G2 is characterized by a higher variance, this is due to the presence of an outlier.

Duration of task phases: We computed the duration associated with T_{ph1}, T_{ph2} and T_{ph3} (Figure 4.2) for each volunteer, thus obtaining the results presented in Figure 4.9. In each graph, the lowest duration of T_{ph1}, T_{ph2} and T_{ph3}, respectively, is highlighted in red. These

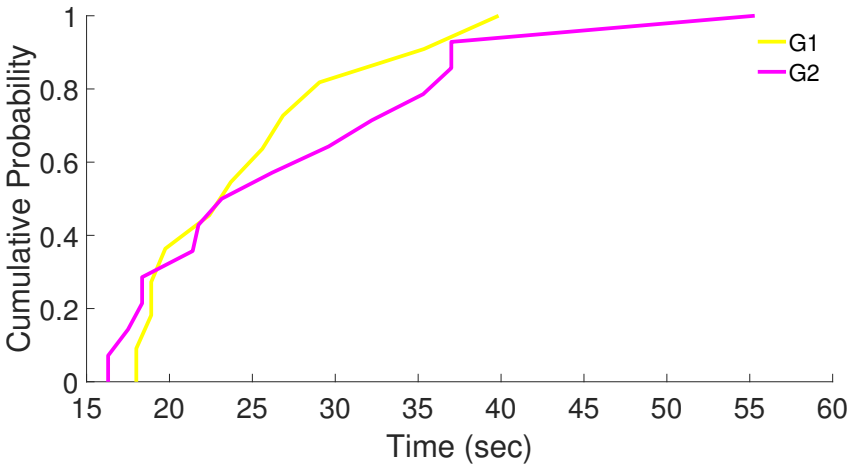


Figure 4.8 The empirical distribution function computed on the time duration of the teaching procedure for each group.

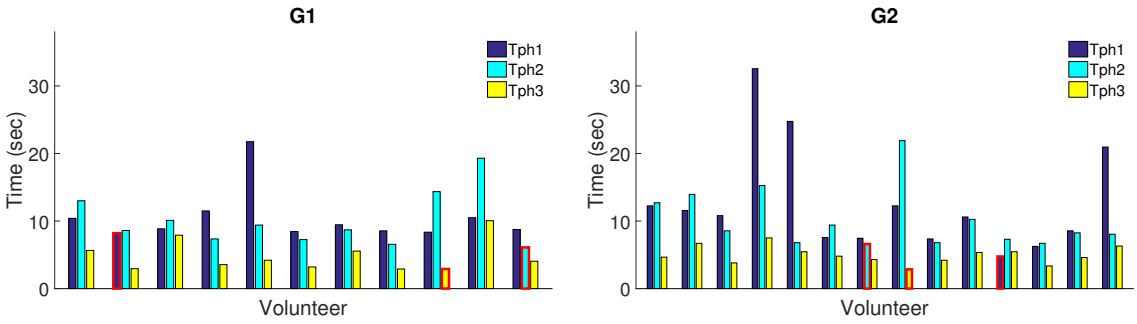
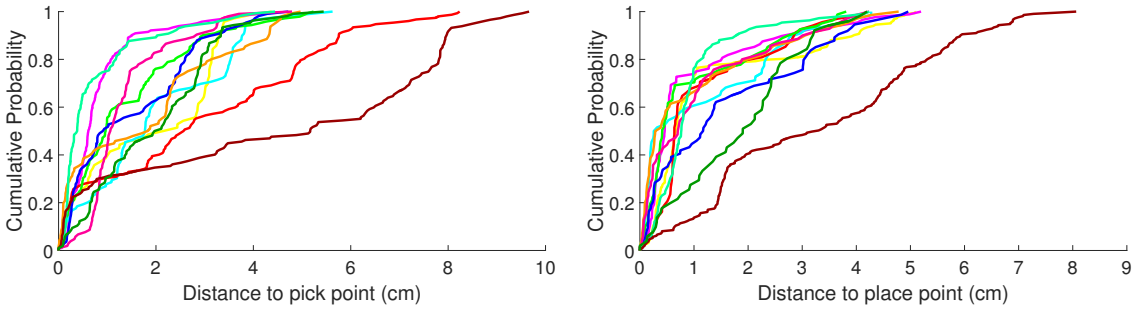
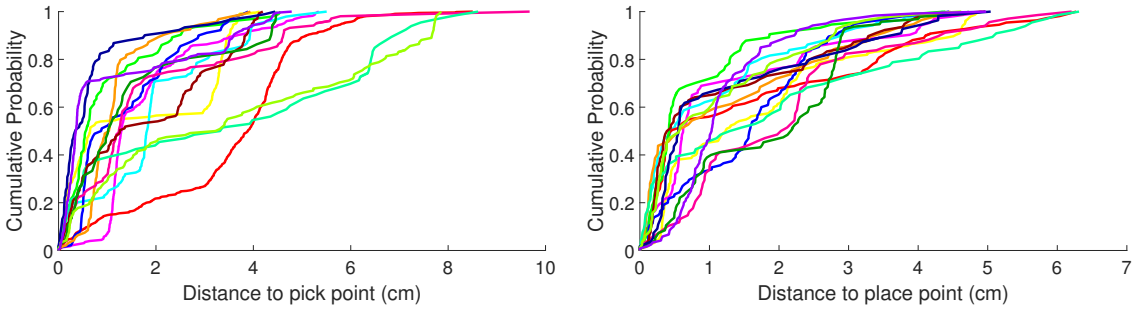


Figure 4.9 Bar graphs, one for each group, representing the time spent in each phase during the teaching procedure. Phases names refer to the temporal classification in Figure 4.2.



(a) Group 1



(b) Group 2

Figure 4.10 The empirical distribution function computed for each volunteer on the distance between points belonging to T_γ and T_r respectively with p_γ and p_r.

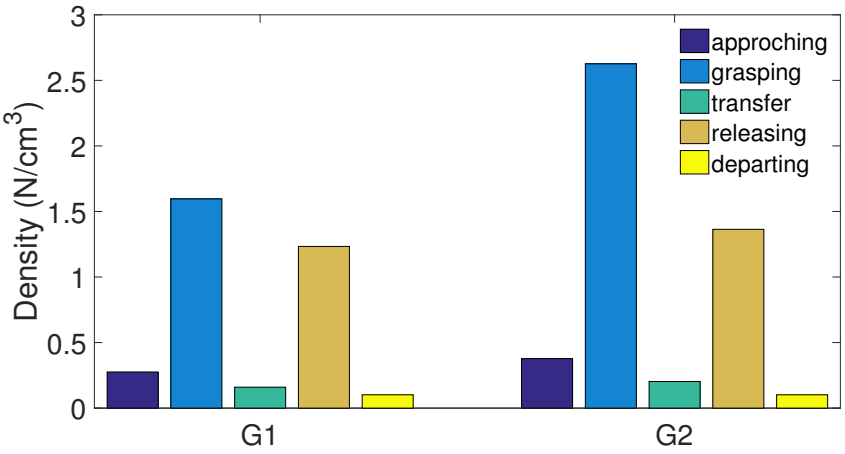
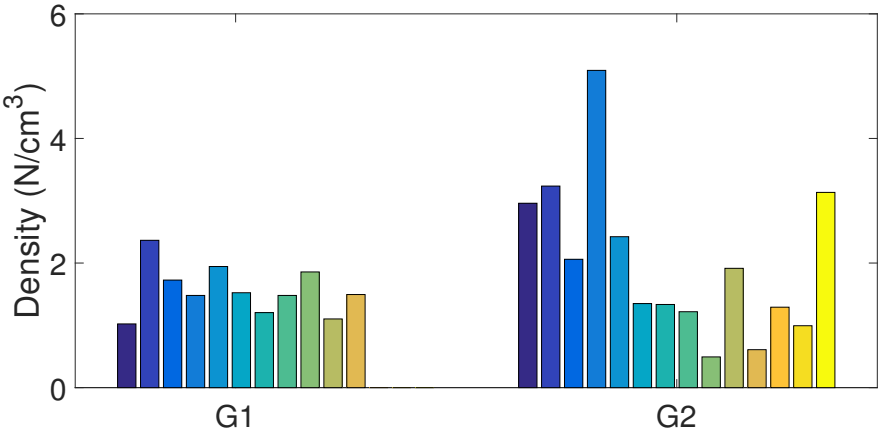
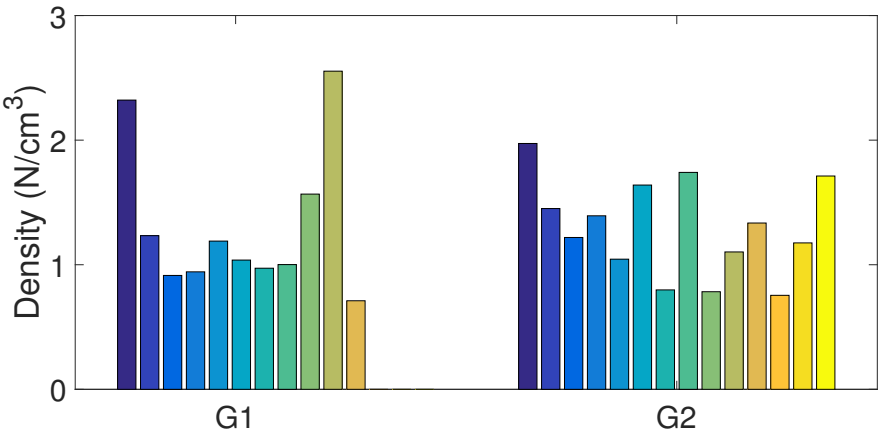


Figure 4.11 Bar graphs, one for each group, representing the median over all the volunteers of the median density for each phase.



(a) *Grasping* phase (T_γ)



(b) *Releasing* phase (T_r)

Figure 4.12 Bar graphs representing the median density for each volunteer in the two critical phases.

results suggest that, as hypothesized in $H_{1.2}$, the time spent by each volunteer on T_{ph1}, T_{ph2} and T_{ph3} is different. As a consequence, if a volunteer optimized the teaching time in one phase this would not imply they would optimize the other two as well.

Spatial distribution of end-effector trajectory points: The density for all the experiments has been computed and used to divide τ into the five intervals expected by the spatial classification as shown in Figure 4.6. Figure 4.10 shows for each group the empirical distribution function computed on the distances between points belonging to T_γ and T_r, respectively with p_γ and p_r. In these graphs we have one line for each experiment where the y-axis indicates the probability for a point to have a distance to the reference point lower than the corresponding value on the x-axis. The main outcome of this study is that, referring to (4.5), ε amounts to 10 *cm* and the majority of the points are far closer to the reference point (p_γ or p_r). Considering the definition of T_γ and T_r given in Section 4.2.2 and the considered setup, as well as the distance between location A and B of 78 cm, the ε value makes it reasonable to consider valid the density-based subdivision.

The bar graph in Figure 4.11 presents the median value of densities over all the experiments for each group. It highlights that the densities of T_γ and T_r are higher with respect to the other phases, confirming H_2, as it was realistic to expect. Figure 4.12 shows the density median value for T_γ and T_r over all the volunteers. These graphs highlight the importance of each operator skills and the uncorrelation of volunteer's performance among phases.

Furthermore, we have computed the time length associated with T_γ and T_r and we have expressed $T_\gamma + T_r$ as percentage over the total teaching time: the resulting median values over all the volunteers are 44.67% for G1 and 45.33% for G2.

Qualitative observations

Videos recorded during the *Human-Human PbD* activities have been inspected by the experimenters to compare differences in teacher behaviours with respect to their behaviour while teaching to Baxter. The most important observation is that the teachers expects the human learner to be active during the training procedure. Contrary to the Baxter-based setup used in these experiments, humans have a context-aware representation of the environment and of the task that is going to be performed and they use this information to predict and anticipate the teacher. In fact, observing Figure 4.13 it is possible to notice that the teacher is just controlling learner's wrist and elbow, nevertheless the learner is able to successfully grasp the box without any instruction on how to position palm and fingers, and on when to actually grasp the object.

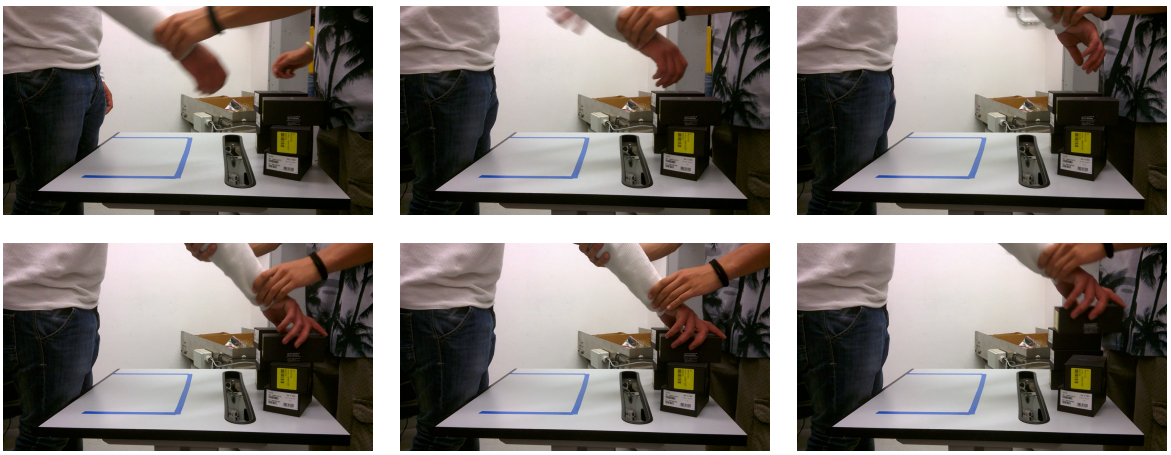


Figure 4.13 *Human-Human PbD* activities: one volunteer is using kinaesthetic teaching to train another person how to pick and place a set of boxes.

Hypotheses corroboration

In Section 4.3.3, when analysing density distributions in trajectories, we found out that phases T_γ and T_r tend to group high density points. Furthermore, observations suggest that, in a typical pick and place task, the time spent in those phases is highly relevant, and amounts to around 45% of the total teaching time. These two findings prove that the process of teaching a robot how to grasp or release an object via KT is critical, and that it would be worth the effort to improve it, since it may lead to a relevant reduction of the required teaching (and possibly execution) time. As we observed in Section 4.3.3 the densities of points in phases T_γ and T_r are characterized by a high variance among different volunteers. This fact is probably due to the difference in their personal skills or lack of experience, and it suggests the necessity to *reduce* an operator's *influence* on certain difficult phases of KT, by extending KT with a number of semi-autonomous robot behaviours. In order to devise such behaviours we recur to *Human-Human PbD* activities, and in particular to what happens when humans apply a simple form of KT to train each other. Qualitative observations of Section 4.3.3, combined with the analyses described in Section 4.3.3, lead us to identify two possible *disturbing factors* in the teaching procedure: (i) the manual control for opening and closing the robot's gripper, as suggested by WH_2, and (ii) the need to select and reach the most appropriate grasping pose, as put forth by WH_3.

In order to avoid these disturbing factors, we propose a research work plan that foresees the design and implementation of two semi-autonomous robot behaviours extending the basic KT paradigm: (i) the autonomous opening and closing of a robot's gripper when appropriate, which requires the robot to understand the operator's intentions and to reason about the

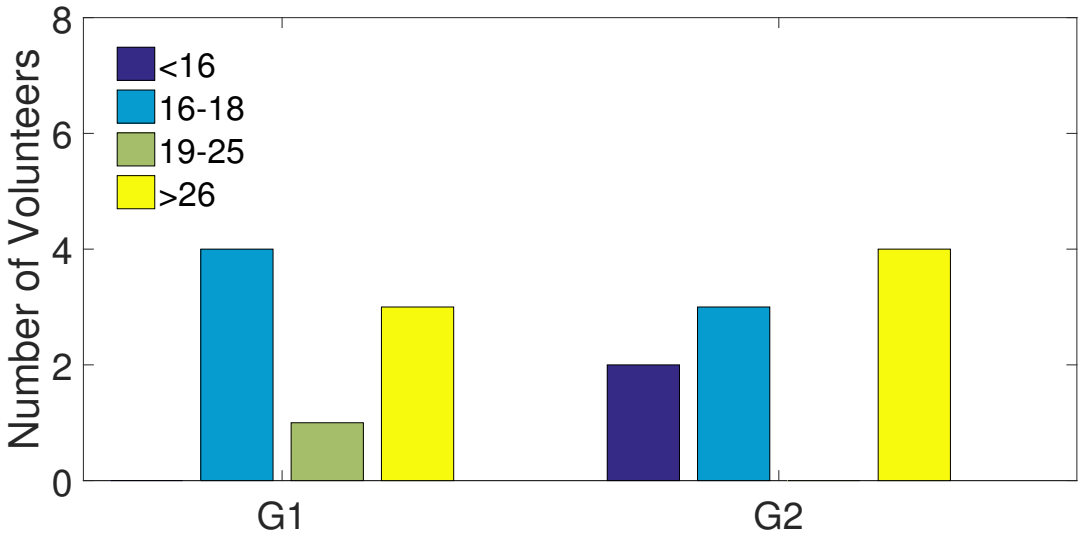


Figure 4.14 A bar graph representing the age distribution between the two groups.

configuration of the objects in its workspace, and (ii) the autonomous gripper re-orientation according to the identified grasping pose, which requires the robot to reason about object shapes and functions. The implementation of such abilities can allow a robot to actively participate in the teaching procedure, reducing the required teaching time and facilitating human-robot interaction via KT. We hypothesize that the introduction of the autonomous opening and closing of the robot's gripper reduces $[H_3]$ the time required to teach a robot a given task, comprehensive of operator's attempts and the required number of iterations. In particular, $[H_{3.1}]$ teaching time variation over volunteers is reduced, and $[H_{3.2}]$ teaching time spent in T_γ and T_r decreases. Instead considering the robot's end-effector trajectory τ, as executed at teaching time, the autonomous gripper opening and closing behaviour $[H_4]$ reduces the density associated to spatial intervals related to *grasp* and *release* instants. In particular, $[H_{4.1}]$ T_γ and T_r variation over volunteers is reduced.

4.3.4 Main study

In order to verify the hypotheses inspired by the *preliminary study* described in Section 4.3.1, we designed a new experiment. The new experimental setup considers just the *Human-Baxter PbD* activity carried on with two different experimental conditions: *Standard KT* (S-KT) and *Wizard of Oz KT* (WoZ-KT).

Volunteers

The experiment involved 17 volunteers, 12 from a vocational educational and training school and 5 from a state industrial and technical institute. Volunteers have been divided into two groups such that the schools are equally represented, Figure 4.14 presents the age of volunteers for the two groups. Two volunteers for each group have previous knowledge of industrial equipment and each of them had between 6 and 12 months of experience. Furthermore, one volunteer for each group has experience with Baxter since he had participated to the preliminary study (Section 4.3.1). The two groups perform the *Human-Baxter PbD* activity with two different experimental conditions: the first group uses S-KT while the second one uses WoZ-KT.

Human-Baxter PbD experiments

The experimental setup for the *Human-Baxter PbD* activity is identical to the one described in Section 4.3.1 but for the position of volunteers, who are beside Baxter. The two groups G1 and G2 perform the experiment with different experimental conditions:

(G1) ***Standard KT:*** The experiment is identical to the one described in Section 4.3.1;

(G2) ***Wizard of Oz KT:*** Volunteers physically guide Baxter through the teaching procedure but they do not have to control the gripper since they are told that Baxter is able to recognize when to grasp and release the object. Instead, an experimenter is controlling the gripper using keyboard commands.

At all times, the trajectory τ is recorded as expressed in the joint space.

Quantitative results

Duration of the teaching process: Figure 4.15 presents the time duration of the teaching procedure for all volunteers, divided by group. The two groups are characterized by different trends. Indeed volunteers in G1 are characterized by a median value of 20.5 *sec* and a variance of 70.26 s^2, while volunteers belonging to G2 have a median value of 12.6 *sec* and a variance of 3.55 s^2. These results together with the empirical cumulative distribution study presented in Figure 4.16 confirm $[H_3]$ and $[H_{3.1}]$, since the autonomous behaviours have reduced both the teaching time mean and variation over volunteers.

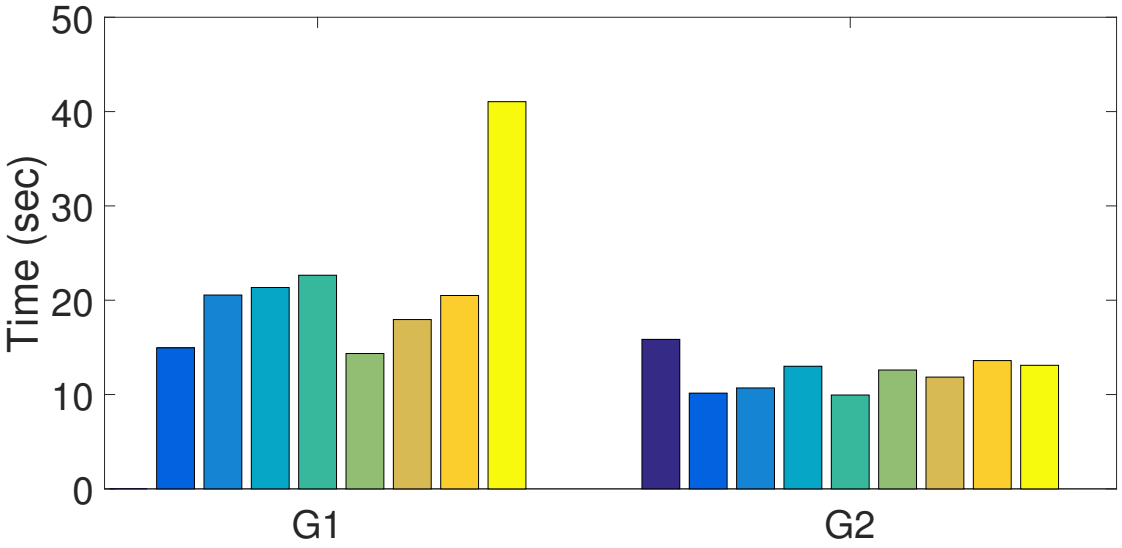


Figure 4.15 The time duration of the teaching procedure in the *Human-Baxter PbD* activities, for each volunteer, in groups.

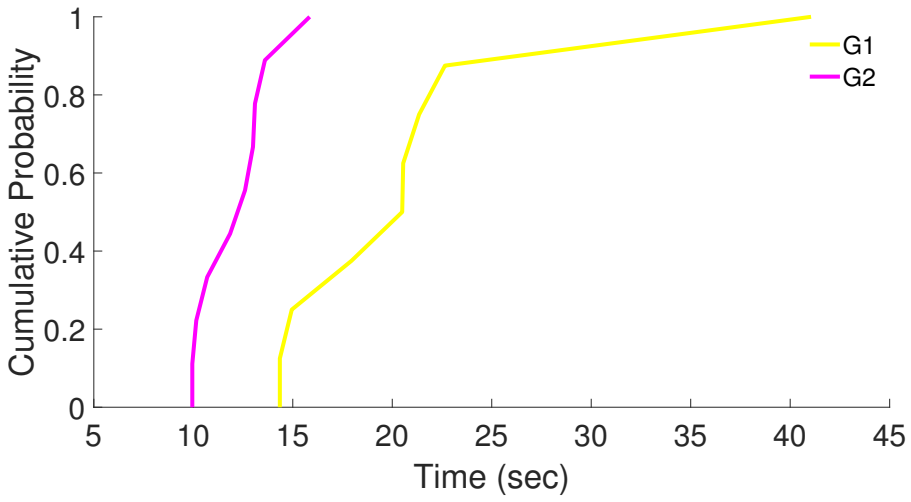


Figure 4.16 The empirical distribution function computed on the time duration of the teaching procedure for each group.

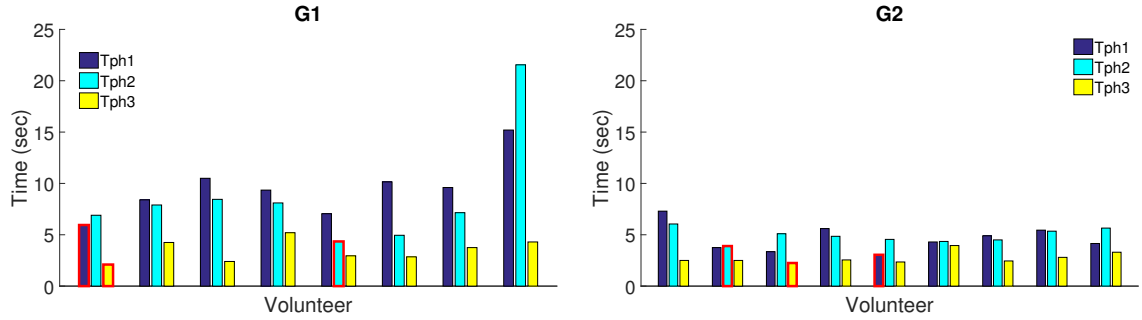
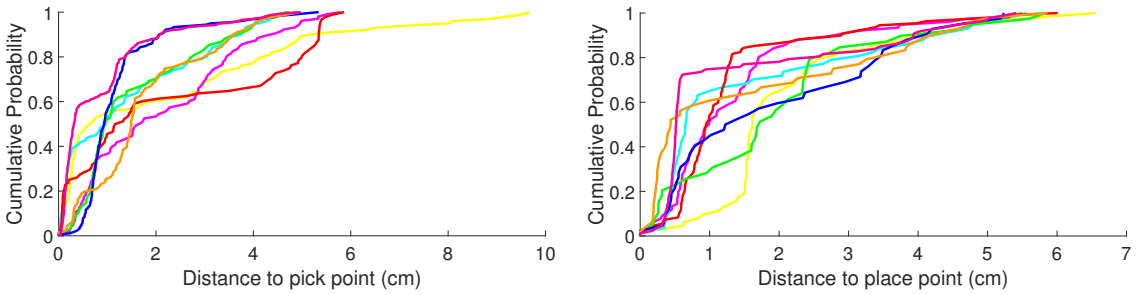
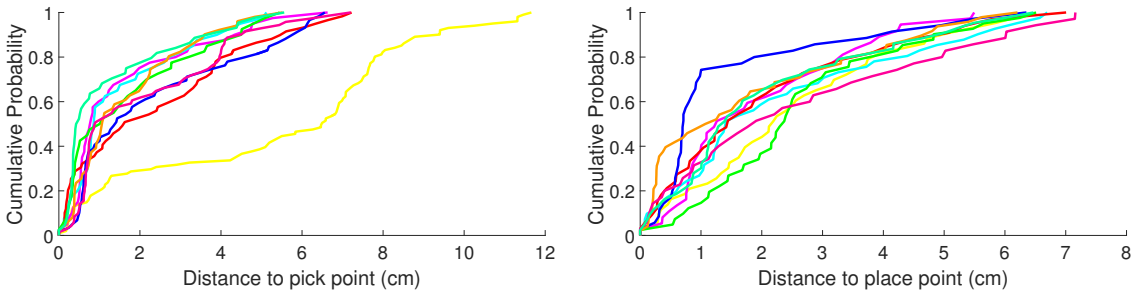


Figure 4.17 Bar graphs, one for each group, representing the time spent in each phase during the teaching procedure. Phases names refer to the temporal classification in Figure 4.2.



(a) Group 1



(b) Group 2

Figure 4.18 The empirical distribution function computed for each volunteer on the distance between points belonging to T_γ and T_r respectively with p_γ and p_r.

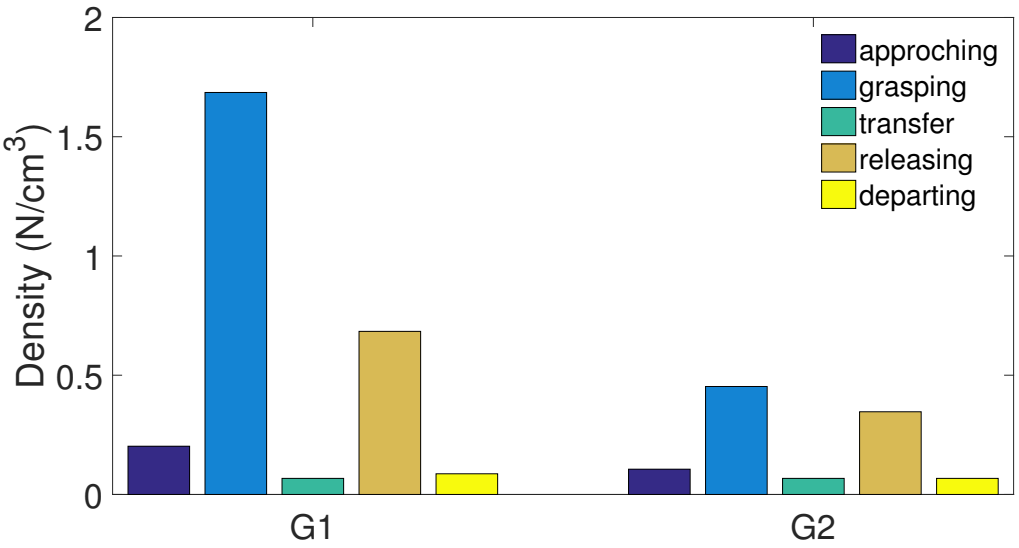


Figure 4.19 Bar graphs, one for each group, representing the median over all the volunteers of the median density for each phase.

Duration of task phases: We computed the duration associated with T_{ph1}, T_{ph2} and T_{ph3} (Figure 4.2) for each volunteer, thus obtaining the results presented in Figure 4.17. Such results reinforce precedent observations regarding H_3. For each group the lowest duration of T_{ph1}, T_{ph2} and T_{ph3} is highlighted in red. These results, analogously to what we found out in the preliminary study, suggests that the time spent by each volunteer on T_{ph1}, T_{ph2} and T_{ph3} is different. We hypothesized that these differences are due to factors ascribable to the volunteer and not to the robot nor the task. In fact, the median value over all the volunteers of the T_{ph1}, T_{ph2} and T_{ph3} variance for G1 is 7.55 s^2 while for G2 is 1.73 s^2. These results seem to confirm that removing the disturbance associated with gripper control increases homogeneity over task phases and volunteers.

Spatial distribution of end-effector trajectory points: The density function $d_{\tau,p}$ is computed for all the experiments with $\Delta = 4.7$ *cm* and it is used to divide each trajectory τ in the five phases defined by the spatial classification. In order to prove the validity of the division process we study the empirical distribution of the distances between points belonging to T_γ and T_r, respectively with p_γ and p_r. From this study, which is summarised in Figure 4.18, results that, referring to (4.5), ε amounts to 12 *cm* while the majority of the points belonging to T_γ and T_r are far closer to the reference points p_γ and p_r. Considering the definition of T_γ and T_r given in Section 4.2.2 and the experimental setup, particularly the 78 *cm* distance between A and B, the determined ε value suggests that the density-based subdivision is valid.

Figure 4.19 presents the median value of densities over all the experiments for each group. Densities for G1 are similar to the one we have found out in the preliminary study (Section 4.3.1). Instead, the densities for G2 show the behaviour that we hypothesized in $[H_4]$, i.e., the autonomous behaviour reduces densities associated with *grasp* and *release* instants. In fact, densities for T_γ and T_r decrease. In particular, the gripper *autonomous* behaviour affects more the *Grasping* phase which is the one characterized by the highest density. The *Grasping* phase is also characterized by a higher variance over volunteers both in the preliminary study and for G1 in the current experiment, particularly for this group the variance is 2.72 N^2/cm^6.Providing the robot with an *autonomous* gripper control (from the volunteers' perspective), as in case of group G2, reduces the density variance among volunteers to 0.01 N^2/cm^6, therefore corroborating $[H_{4.1}]$.

Using the density-based segmentation of τ we have computed the time length associated to T_γ and T_r and we have expressed $T_\gamma + T_r$ as a percentage over the total teaching time. The results seem to confirm $[H_{3.2}]$ since for G1 the two phases take 43% of the total time while for G2 the total time taken by this two phases reduces to 37%.

4.3.5 Discussion

In Section 4.3.3 discussing the results of the preliminary study, we concluded that phases T_γ and T_r could benefit by two semi-autonomous behaviours: namely the autonomous opening and closing of the robot's gripper, inspired by WH_2, and the autonomous gripper re-orientation according to an identified grasping pose, inspired by WH_3. Focusing on the first semi-autonomous behaviour we formulated a new set of hypotheses and we structured a new experiment to test them.

The proposed semi-autonomous approach is in line with the tendency of previous works toward a more natural teaching modality such as data gloves [149], vision systems [150], haptic devices [151] or kinaesthesia [152]. While tactile corrections have been proposed to refine demonstrations which may be negatively affected by human inexperience [147], our approach reduces the effects introduced by operators, as shown in Section 4.3.4, by limiting their responsibilities according to what we observed during the natural human-human interaction.

The *Wizard of Oz* approach for its nature cannot provide a definitive answer to our research questions since the experimenter influence can be neither neglected nor estimated. If it is reasonable to assume that the experimenter is more accurate than an autonomous system

in detecting *grasp* and *release* instants, it is not realistic to assume that they have the same repeatability. Nevertheless, the results of this study can provide useful hints.

Our results seem to confirm our hypothesis about the suboptimalities introduced in the teaching procedure and the simulation of the semi-autonomous behaviour partially solves the criticality introduced by *grasping* and *releasing*. The autonomous gripper re-orientation according to the identified grasping pose could be what is missing in order to obtain a homogeneous teacher-independent teaching process free from non-intuitive routine operations.

4.4 Follow Up

[5] The results of our experiments with PbD and KT suggested us that an autonomous behaviour of the robot in the training phase could help to improve the overall quality of the trajectory and to reduce the teaching time. As we have pointed out in Section 4.1 the movements performed by a human for KT are continuous gestures that modify the robot joint status. Since normally this does not involve the usage of any other external sensors the visibility of the KT gestures is at the robot joint level and therefore they are sensed using the robot encoders. Because of this reason is not possible to determine if the user is using both hands (bi-manual gesture) or just one. KT gestures clearly have a direct effect on all the robot joints except for the gripper, and this is probably what causes the problems experienced in our studies. For this reason, we think that a reasonable approach is to link the KT gestures to the gripper control to implement our desired autonomous behaviour.

To achieve this we have decided to use a data-driven approach similar to those described in Chapter 3. A small dataset (36 sequences) have been collected with two volunteers teaching pick and place tasks to Baxter using KT. Each volunteer has been asked to use KT to pick an object 18 times, every time from a different spatial position, using only one hand. The robot starts every time from its default configuration. Volunteers wear a smartwatch on the hand used to control the robot arm. The overall dataset contains smartwatch angular velocity and linear acceleration, joint state (angles, velocities and efforts) for the controlled arm and the gripper status. Using the sequences contained in the dataset an LSTM neural network has been trained to map the current robot status and human data from the smartwatch to the robot end-effector status. In this preliminary study, the neural network has been trained only for the grasping scenario. To perform online testing, the data collected by the

[5] The work described in this section has been carried out by Carlo Adornetto and supervised by Alessandro Carfì for the course Software Architecture for Robotics


Figure 4.20 Two frames extracted from the video representing two volunteers using KT with the developed autonomous behaviour.

system are fed to the neural network through a moving horizon window and the output is continuously sent as a control command to the robot gripper. Results of this preliminary study seem encouraging since the robot is able to autonomously close the gripper during the teaching phase, Figure 4.20 presents two frames extracted by a video where two volunteers use the system [6]. Obviously the dataset is too small and the experiments limited to extract final considerations. Moreover, the model should be extended to consider also the releasing scenario.

[6]https://drive.google.com/file/d/1DIwU1Snii_FKS4cgX_TTy3X4_TKTQgjo/view?usp=sharing

Chapter 5

Future Challenges & Conclusions

In the previous Chapters we have investigated the GI-UI problem in robotics scenarios focusing on two specializations of the problem Discrete GI-UI, Chapter3, and Continuous GI-UI, Chapter 4. This leads us to develop techniques and design experimental setups to deal with discrete and continuous gestures. The developed technologies and the acquired experience, for the GI-UI problem, can be used in other application fields as a natural evolution of our work.

5.1 Activity Recognition

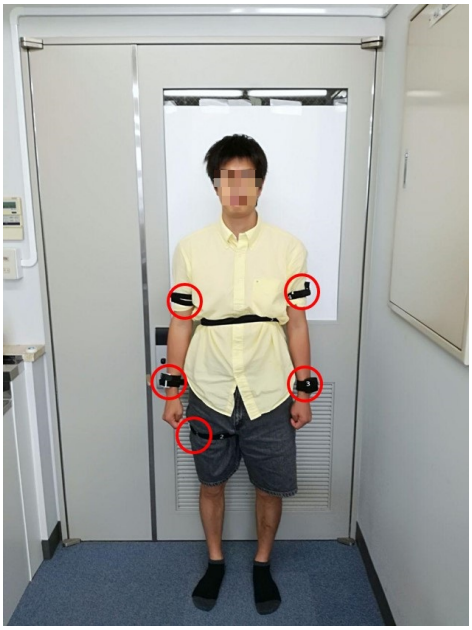
[1] As we have mentioned in Chapter 3 the gesture and activity recognition problem shares some come aspect such as sensing devices and processing techniques. Therefore, the techniques developed for gesture recognition have been adapted and tested for the activity recognition problem.

Activity recognition is a widespread research team with different application scenario. In this study, we were interested to recognize activities of daily leaving (ADL) in an healthcare scenario. The idea is that by monitoring some activities such as drinking, walking or brushing the teeth it is possible to infer the health status of the subject and its level of autonomy. An intelligent system capable of recognizing that you are sitting for a long time can suggest or encourage some physical activity, similarly, if the drinking activity has not been recognized for a while a reminder could be triggered to prevent dehydration. Of course, the core functionality for a system like this to properly work is the activity recognition. Since these

[1]The work described in this section has been carried out by Marco Ruzzon and supervised by Fulvio Mastrogiovanni, Toshiyuki Murakami and Carfì Alessandro for the Master Thesis "A Modular Architecture for Activity Recognition using Wearable Devices and Neural Networks"

Table 5.1 Activities considered in the study.

ID	Activities	ADL Category	ADL Description
1 2 3 4 5	Walk Sit Down (chair) Stand Up (chair) Open Door Close Door	Transferring/Mobility (Basic)	moving oneself from seated to standing, getting in and out of bed, and the ability to walk independently from one location to another
6 7	Pour Water Drink Glass	Eating (Basic)	the ability to feed oneself, though not necessarily the capability to prepare food
8	Brush Teeth	Personal Hygiene (Basic)	bathing/showering, grooming, nail care and oral care
9	Clean Table	Housework (Instrumental)	cleaning and maintaining the house



(a) Sensor Configuration (Front)

(b) Sensor Configuration (Back)

Figure 5.1 Sensors position on the volunteer body.

systems are usually meant to be deployed in areas associated with high concerns for privacy, such as a private apartment, cameras are usually avoided in favour of wearables.

Wearables guarantee a higher level of privacy and can be easily embedded in objects of daily usage such as wristbands, watches and more generally cloths. Ideally, a distributed architecture can be imagined where the sensors are worn in different body location to enhance the collected information. However, wearable sensors require power delivery

through batteries and given the scenario is not reasonable to assume that the subject is going to personally charge all the devices when needed. Therefore, we are looking for a flexible and modular software architecture that can continue to operate even if not all the sensors are active, waiting for a caregiver to charge the sensing devices.

With these premises, we are looking for a distributed architecture, capable of recognizing different activities using data from different sensors and with a modular structure capable of handling sensors failure. The SPC technique (Section 3.6.2), developed to overcome the lack of modularity of SLOTH, fits the system requirements.

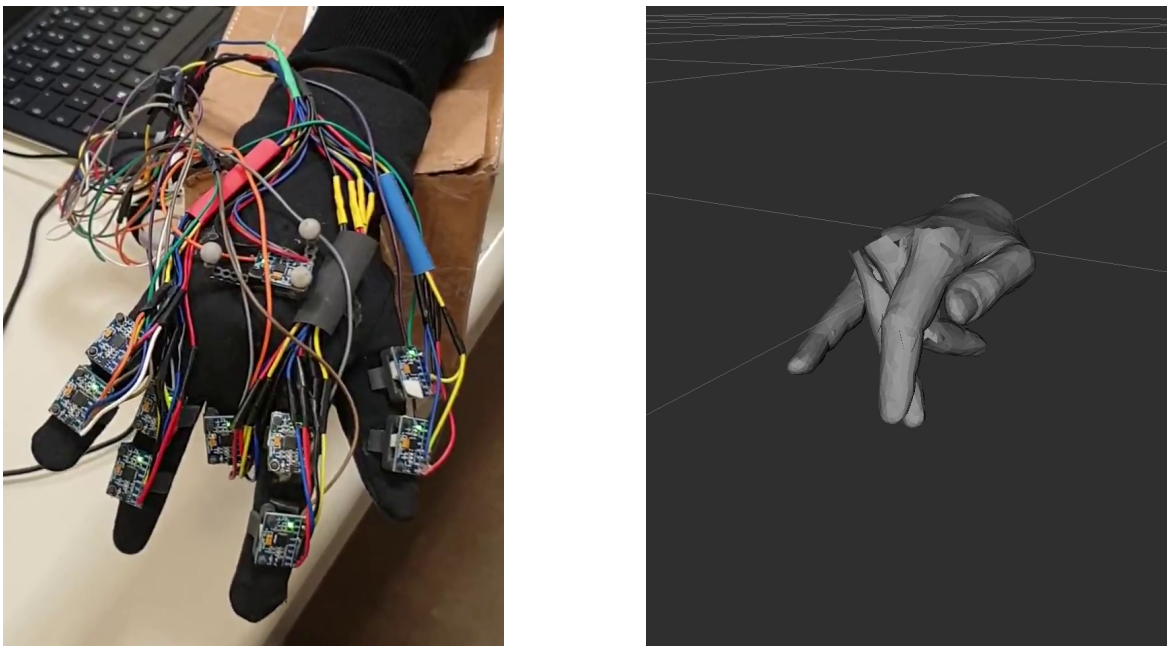
Therefore, we designed a software architecture based on SPC to recognize the ADL described in Table 5.1 using 6 9-axis IMUs worn as shown in Figure 5.1. Data from each of the 6 sensors are processed by an independent module. Each module contains a predictor for each activity, implemented using LSTM-NN, returning a continuous stream of prediction error that is processed by a threshold mechanism to perform the detection. Recognition result from each sensor module is constantly sent to a local reasoner that uses a voting system to determine the performed activity. Each module has been evaluated individually and the preliminary results on the overall architecture are promising.

5.2 Hand Tracking

InDex project [2] aims to study human in-hand manipulation for robotic purposes. Autonomous robots are expected to operate in human environment and physically interact with objects designed for humans, objects that should be manipulated according to their physical characteristics such as fragility and affordance. In literature, many works have been proposed to extract information from human motion to learn strategies for in-hand manipulation for robotic hands. In particular, multimodal data from human interaction with objects have been proposed to learn how to grasp and manipulate objects [163], data from a smart glove, equipped with magnetic trackers, have been used to learn grasp strategies [164] and human demonstrations, recorded in virtual reality, have been used to improve reinforcement learning results in manipulation tasks [165]. Human motion tracking consist in estimating at each time instant the status of the human joints ($\boldsymbol{q}(t)$ from Section 2.1) and it is required to collect good quality data from which a robot could learn. To learn manipulation strategies, hand tracking is fundamental.

[2]The work described in this section have been founded from the European Commission within the Horizon 2020 Framework (CHIST-ERA,2014-2020), project InDex.

In Section 3.6.1 we have described a data glove equipped with 12 6-axis IMUs to perceive human gestures. In our work, we have explored how to use IMUs raw data to interpret human motion without using any kind of motion tracking. Nevertheless, IMUs can even be used for motion tracking, e.g. Xsense MoCap [3].



(a) The real hand pose (b) The perceived hand pose

Figure 5.2 Hand tracking example using the data glove.

The sensors embedded in our data glove uses raw data to extract the sensor orientation expressed in quaternions. Since each sensor is mechanically coupled with a hand or finger link, the difference in orientation between consecutive links can be used to determine the hand joint states. However, the orientation information is extracted from the sensory data (accelerometer and gyroscope) and it is particularly accurate in estimate the sensor roll and pitch but, since the sensor is missing a magnetometer, the yaw estimation is performed integrating the gyroscope data without an external reference. This makes the yaw estimation particularly sensitive to noise and initial state conditions. Therefore, to compute the angle between two joints is not possible to simple relay on the rotation matrix between the two consecutive sensor frames, since their orientation differences are influenced by errors in the yaw estimation and not only by the human motion. To overcome this problem we use the orientation information to define a plane for each sensor, perpendicular to the z-axis and

[3]https://www.xsens.com/motion-capture

therefore independent by the yaw. Then joint angles are computed as the angle between planes related to consecutive sensors. This led to good tracking result shown in Figure 5.2 although quantitative results, comparing the glove tracking with a vision based MoCap system, should still be carried out.

5.3 Conclusions

In this thesis work, we have faced the problem of GI-UI, not necessarily aiming to a natural communication tool, but a new tool leveraging a communication channel that nowadays, with some exceptions, is mostly unused in mainstream technology. We have explored the literature realizing that a big obstacle in this technology development the lack of a clear definition of the problem and confusion in keywords used to refer it. To structure the problem we have given a formal definition of gesture and we have designed a gesture taxonomy. Furthermore, we divided the problem of gestural interaction in three components sensing, data processing and system reaction and we have analysed implications of choices taken for each component. Since each component is strictly correlated with the others, it is not possible to address them individually but a more homogeneous approach should be adopted.

The proposed taxonomy and the division of the gestural interaction problem have been used to classify articles in the context of a literature survey. The classification is presented in a tabular form, easy to be consulted, and an online version has been made available as well. The website solution is an efficient way to constantly update the survey and to let other researchers cooperate in this process. We hope that this could help in a further field development by building a research community sharing the same terminologies. Our literature classification gives the possibility to researchers to look for relevant works filtering them for specific aspects such as sensing device, problem faced or the kind of gesture considered. This is what we have done in Chapter 3, we have chosen the filtering criteria, such as the usage of discrete gestures and IMUs, based on our application scenario, and we have extracted from our survey relevant articles.

In Chapter 3, we have tackled the Discrete GI-UI problem exploring the usage of discrete gestures for human-robot interaction to navigate a menu-based interface through the usage of wearable IMUs for an industrial application. We have designed a new gesture recognition method named SLOTH that leverages Long Short Term Memory Neural Network and a threshold mechanism to perform online gesture recognition relaxing the close world assumption. We have compared SLOTH with another state of the art method that uses Gaussian Mixture Modelling and Gaussian Mixture Regression to build gesture templates, the

Mahalanobis distance to perform a probabilistic classification and a thresholding mechanism for gesture detection. The comparative study results highlight SLOTH higher performances both in term of gesture recognition accuracy and system responsiveness. Using SLOTH for the gesture recognition we have developed a menu-based interface to interact with a CoBot and we have designed and performed a subject study. Results of this studies identified problems in the gesture dictionary and more generally in the dataset. Therefore, a list of activities has been planned to overcome these problems. In particular, we have designed and built a data glove for gesture sensing, designed a new gesture dictionary and develop a new protocol and software to collect a new dataset. Furthermore, a new gesture recognition method has been envisioned (Simultaneous Prediction and Classification SPC) to overcome SLOTH limitations for what regards system modularity. Future development of this work will consist in an assessment of the new dictionary either for intuitiveness and physical stress, the collection of a bigger dataset and a more extensive evaluation of the proposed gesture recognition methods, SLOTH and SPC.

Finally, in Chapter 4 we have considered the Continuous GI-UI problem studying the usage of continuous gestures associated with the programming by demonstration paradigm, known as Kinaesthetic Teaching (KT). We have conducted a preliminary experiment to study the trajectories performed by the user during the KT of a pick-and-place task. The preliminary experimentation underlined the criticality of the grasping and releasing phases and suggested that an active robot behaviour, during the teaching phase, could facilitate the user work. With a follow-up study, we have enforced this conclusion thanks to a Wizard of Oz experiment. Therefore, we have applied the expertise in data-driven approaches, developed for the discrete gesture recognition, to implement an autonomous grasping behaviour for the KT obtaining good preliminary results. However, this work should be consolidated and extended for the releasing phase aiming to a shift in the KT in which the robot is no more a passive subject but it is actively able to interact. This can be achieved by developing better tools for reacting to continuous gestures and probably by adding some context information in the loop.

We hope that the influence of our work could exceed our expectations inspiring the work of researchers in gestural interaction and similar fields.

References

[1] Henning Kagermann, Johannes Helbig, Ariane Hellinger, and Wolfgang Wahlster. *Recommendations for implementing the strategic initiative INDUSTRIE 4.0: Securing the future of German manufacturing industry; final report of the Industrie 4.0 Working Group*. Forschungsunion, 2013.

[2] Dominik Lucke, Carmen Constantinescu, and Engelbert Westkämper. Smart factory - a step towards the next generation of manufacturing. In *Manufacturing Systems and Technologies for the New Frontier*, pages 115–118. Springer, 2008.

[3] Sami Haddadin, Alin Albu-Schaffer, Alessandro De Luca, and Gerd Hirzinger. Collision detection and reaction: A contribution to safe physical human-robot interaction. In *Procceding of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3356–3363, Nice, France, October 2008.

[4] Alessandro Albini, Simone Denei, and Giorgio Cannata. Enabling natural human-robot physical interaction using a robotic skin feedback and a prioritized tasks robot control architecture. In *Proceedings of the 17th IEEE/RAS International Conference on Humanoid Robotics (Humanoids)*, pages 99–106, Birmingham, UK, November 2017.

[5] Przemyslaw A Lasota, Gregory F Rossano, and Julie A Shah. Toward safe close-proximity human-robot interaction with standard industrial robots. In *Proceeding of the IEEE International Conference on Automation Science and Engineering (CASE)*, pages 339–344, Taipei, Taiwan, October 2014.

[6] Kourosh Darvish, Francesco Wanderlingh, Barbara Bruno, Enrico Simetti, Fulvio Mastrogiovanni, and Giuseppe Casalino. Flexible human–robot cooperation models for assisted shop-floor tasks. *Mechatronics*, 51:97–114, 2018.

[7] Aude Billard, Sylvain Calinon, Ruediger Dillmann, and Stefan Schaal. Robot programming by demonstration. *Springer Handbook of Robotics*, pages 1371–1394, 2008.

[8] J Norberto Pires. Robot-by-voice: Experiments on commanding an industrial robot using the human voice. *Industrial Robot*, 32(6):505–511, 2005.

[9] Ashish Singh, Stela H Seo, Yasmeen Hashish, Masayuki Nakane, James E Young, and Andrea Bunt. An interface for remote robotic manipulator control that reduces task load and fatigue. In *Proceedings of the 22th IEEE International Workshop on Robot and Human Interactive Communication (RO-MAN)*, pages 738–743, August 2013.

[10] Justine Cassell. A Framework for Gesture Generation and Interpretation. In *Computer Vision for Human–Machine Interaction*, pages 191–216. 2010.

[11] Anand G Buddhikot, Nitin M Kulkarni, and Arvind D Shaligram. Hand Gesture Interface based on Skin Detection Technique for Automotive Infotainment System. *Image, Graphics and Signal Processing*, 2(11):10–24, 2018.

[12] David Kortenkamp, Eric Huber, and R. Peter Bonasso. Recognizing and interpreting gestures on a mobile robot. In *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI)*, Portlan, Oregon, USA, August 1996.

[13] Yoshinori Kuno, Teruhisa Murashina, Nobutaka Shimada, and Yoshiaki Shirai. Intelligent wheelchair remotely controlled by interactive gestures. In *Proceedings of the 15th International Conference on Pattern Recognition (ICPR)*, volume 4, pages 672–675, Barcelona, Spain, September 2000.

[14] Md Hasanuzzaman, V Ampornaramveth, Tao Zhang, M A Bhuiyan, Y Shirai, and H Ueno. Real-time Vision-based Gesture Recognition for Human Robot Interaction. In *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Shenyang, China, October 2004.

[15] Pedro Neto, J. Norberto Pires, and A. Paulo Moreira. Accelerometer-based control of an industrial robotic arm. In *Proceedings of the 18th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 1192–1197, Toyama, Japan, September 2009.

[16] Xing-Han Wu, Mu-Chun Su, and Pa-Chun Wang. A hand-gesture-based control interface for a car-robot. In *Proceeding of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4644–4648, Taipei, Taiwan, October 2010.

[17] Salvatore Iengo, Silvia Rossi, Mariacarla Staffa, and Alberto Finzi. Continuous gesture recognition for flexible human-robot interaction. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4863–4868, Hong Kong, China, May 2014.

[18] Grazia Cicirelli, Carmela Attolico, Cataldo Guaragnella, and Tiziana D'Orazio. A Kinect-Based Gesture Recognition Approach for a Natural Human Robot Interface. *International Journal of Advanced Robotic Systems*, 12(3):22, March 2015.

[19] Yuhui Lai, Chen Wang, Yanan Li, Shuzhi Sam Ge, and Deqing Huang. 3D pointing gesture recognition for human-robot interaction. In *Proceedings of the Chinese Control and Decision Conference (CCDC)*, pages 4959–4964, Yinchuan, China, May 2016.

[20] Enrique Coronado, Jessica Villalobos, Barbara Bruno, and Fulvio Mastrogiovanni. Gesture-based robot control: Design challenges and evaluation with humans. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2761–2767, Singapore, May 2017.

[21] Gabriele Bolano, Atanas Tanev, Lea Steffen, Arne Roennau, and Ruediger Dillmann. Towards a Vision-Based Concept for Gesture Control of a Robot Providing Visual Feedback. In *Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 386–392, Kuala Lumpur, Malaysia, December 2018.

[22] Md Jahidul Islam, Marc Ho, and Junaed Sattar. Understanding human motion and gestures for underwater human–robot collaboration. *Journal of Field Robotics*, 36(5):851–873, August 2019.

[23] Simone Denei, Fulvio Mastrogiovanni, and Giorgio Cannata. Towards the creation of tactile maps for robots and their use in robot contact motion control. *Robotics and Autonomous Systems*, 63:293–308, 2015.

[24] Alessio Capitanelli, Marco Maratea, Fulvio Mastrogiovanni, and Mauro Vallati. On the manipulation of articulated objects in human–robot cooperation scenarios. *Robotics and Autonomous Systems*, 109:139–155, 2018.

[25] Richard A. Bolt. "Put-that-there": Voice and gesture at the graphics interface. In *Proceedings of the 7th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pages 262–270, New York, USA, July 1980.

[26] O. Rogalla, M. Ehrenmann, R. Zöllner, R. Becher, and R. Dillmann. Using gesture and speech control for commanding a robot assistant. In *Proceedings of the 11th IEEE International Workshop on Robot and Human Interactive Communication (RO-MAN)*, pages 454–459, Berlin, Germany, September 2002.

[27] Alessandro Carfì, Carola Motolese, Barbara Bruno, and Fulvio Mastrogiovanni. Online human gesture recognition using recurrent neural networks and wearable sensors. In *Proceeding of the 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 188–195, Nanjing, China, August 2018. IEEE.

[28] Alessandro Carfì, Jessica Villalobos, Enrique Coronado, Barbara Bruno, and Fulvio Mastrogiovanni. Can human-inspired learning behaviour facilitate human-robot interaction? *International Journal of Social Robotics*, pages 1–14, 2019.

[29] Vladimir I. Pavlovic, Rajeev Sharma, and Thomas S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:677–695, 1997.

[30] Nurettin Çağrı Kılıboz and Uğur Güdükbay. A hand gesture recognition technique for human–computer interaction. *Journal of Visual Communication and Image Representation*, 28:97–104, April 2015.

[31] Sushmita Mitra and Tinku Acharya. Gesture recognition: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(3):311–324, 2007.

[32] Hao Tang, Hong Liu, Wei Xiao, and Nicu Sebe. Fast and robust dynamic hand gesture recognition via key frames extraction and feature fusion. *Neurocomputing*, 331:424–433, February 2019.

[33] Gonzalo Pomboza-Junez, Juan A. Holgado-Terriza, and Nuria Medina-Medina. Toward the gestural interface: comparative analysis between touch user interfaces versus gesture-based user interfaces on mobile devices. *Universal Access in the Information Society*, 18(1):107–126, March 2019.

[34] Biplab Ketan Chakraborty, Debajit Sarma, M.K. Bhuyan, and Karl F. MacDorman. Review of constraints on vision-based gesture recognition for human–computer interaction. *IET Computer Vision*, 12(1):3–15, February 2018.

[35] Hyun Kang, Chang Woo Lee, and Keechul Jung. Recognition-based gesture spotting in video games. *Pattern Recognition Letters*, 25(15):1701–1714, November 2004.

[36] Simon Ruffieux, Denis Lalanne, Elena Mugellini, and Omar Abou Khaled. Gesture recognition corpora and tools: A scripted ground truthing method. *Computer Vision and Image Understanding*, 131:72–87, February 2015.

[37] Juan C. Núñez, Raúl Cabido, Juan J. Pantrigo, Antonio S. Montemayor, and José F. Vélez. Convolutional Neural Networks and Long Short-Term Memory for skeleton-based human activity and hand gesture recognition. *Pattern Recognition*, 76:80–94, April 2018.

[38] Chao Hu, Max Qinghu Meng, Peter Xiaoping Liu, and Xiang Wang. Visual gesture recognition for human-machine interface of robot teleoperation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 2, pages 1560–1565, Las Vegas, Nevada, USA, December 2003.

[39] Maria Karam and M. C. Schraefel. A Taxonomy of Gestures in Human Computer Interactions. *ACM Transactions on Computer-Human Interactions*, 2005.

[40] Helman I. Stern, Juan P. Wachs, and Yael Edan. Designing hand gesture vocabularies for natural interaction by combining psycho-physiological and recognition factors. *International Journal of Semantic Computing*, 02(01):137–160, March 2008.

[41] Rajeshri R. Itkarkar and Anilkumar V. Nandi. A survey of 2D and 3D imaging used in hand gesture recognition for human-computer interaction (HCI). In *Proceedings of the IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE)*, pages 188–193, Pune, Maharashtra, India, December 2016.

[42] Feng Jiang, Shengping Zhang, Shen Wu, Yang Gao, and Debin Zhao. Multi-layered Gesture Recognition with Kinect. *The Journal of Machine Learning Research*, 16(1):227–254, July 2017.

[43] Micael Carreira, Karine Lan Hing Ting, Petra Csobanka, and Daniel Gonçalves. Evaluation of in-air hand gestures interaction for older people. *Universal Access in the Information Society*, 16(3):561–580, August 2017.

[44] Francis KH Quek. Eyes in the interface. *Image and Vision Computing*, 13(6):511–525, 1995.

[45] David McNeill. *Hand and mind: What gestures reveal about thought*. University of Chicago press, 1992.

[46] Francis Quek, David Mcneill, Robert Bryll, Susan Duncan, Xin-Feng Ma, Cemil Kirbas, Karl E. Mccullough, and Rashid Ansari. Multimodal Human Discourse: Gesture and Speech. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 9(3):171–193, 2002.

[47] Florent Taralle, Alexis Paljic, Sotiris Manitsaris, Jordane Grenier, and Christophe Guettier. Is Symbolic Gestural Interaction Better for the Visual Attention? *Procedia Manufacturing*, 3:1060–1065, 2015.

[48] Tijana Vuletic, Alex Duffy, Laura Hay, Chris McTeague, Gerard Campbell, and Madeleine Grealy. Systematic literature review of hand gestures used in human computer interaction interfaces. *International Journal of Human-Computer Studies*, 129:74–94, September 2019.

[49] Edgar Rojas-Munoz and Juan Pablo Wachs. MAGIC: A Fundamental Framework for Gesture Representation, Comparison and Assessment. In *Proceedings of the 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG)*, pages 1–8, Lile, France, May 2019.

[50] Juha Kela, Panu Korpipää, Jani Mäntyjärvi, Sanna Kallio, Giuseppe Savino, Luca Jozzo, and Sergio Di Marca. Accelerometer-based gesture control for a design environment. *Personal and Ubiquitous Computing*, 10(5):285–299, August 2006.

[51] Jungpil Shin and Cheol Min Kim. Non-Touch Character Input System Based on Hand Tapping Gestures Using Kinect Sensor. *IEEE Access*, 5:10496–10505, May 2017.

[52] Divya Shah, Ernesto Denicia, Tiago Pimentel, Barbara Bruno, and Fulvio Mastrogiovanni. Detection of bimanual gestures everywhere: Why it matters, what we need and what is missing. *Robotics and Autonomous Systems*, 99:30–49, 2018.

[53] Wenbing Zhao, Deborah D Deborah, M Ann Reinthal, Beth Ekelman, Glenn Goodman, and Joan Niederriter. Privacy-aware human motion tracking with realtime haptic feedback. In *Proceedings of the IEEE International Conference on Mobile Services (MS)*, pages 446–453, New York, USA, June 2015.

[54] Juan Pablo Wachs, Mathias Kölsch, Helman Stern, and Yael Edan. Vision-based hand-gesture applications. *Communications of the ACM*, 54(2):60, February 2011.

[55] Stergios Poularakis and Ioannis Katsavounidis. Low-complexity hand gesture recognition system for continuous streams of digits and letters. *IEEE transactions on cybernetics*, 46(9):2094–2108, 2015.

[56] Hongyi Liu and Lihui Wang. Gesture recognition for human-robot collaboration: A review. *International Journal of Industrial Ergonomics*, 68:355–367, March 2018.

[57] Hong Cheng, Lu Yang, and Zicheng Liu. Survey on 3D Hand Gesture Recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(9):1659 – 1673, September 2016.

[58] MA Brodie, Alan Walmsley, and Wyatt Page. The static accuracy and calibration of inertial measurement units for 3d orientation. *Computer Methods in Biomechanics and Biomedical Engineering*, 2008.

[59] Stephen P Tseng, Wen-Lung Li, Chih-Yang Sheng, Jia-Wei Hsu, and Chin-Sheng Chen. Motion and attitude estimation using inertial measurements with complementary filter. In *Proceedings of the 8th Asian Control Conference (ASCC)*, pages 863–868, Kaohsiung, Taiwan, May 2011. IEEE.

[60] Renqiang Xie, Xia Sun, Xiang Xia, and Juncheng Cao. Similarity Matching-Based Extensible Hand Gesture Recognition. *IEEE Sensors Journal*, 15(6):3475–3483, June 2015.

[61] Alessandro Carfi, Carola Motolese, Barbara Bruno, and Fulvio Mastrogiovanni. Online human gesture recognition using recurrent neural networks and wearable sensors. In *27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 188–195, Nanjing,China, August 2018. IEEE.

[62] Chinmaya R. Naguri and Razvan C. Bunescu. Recognition of Dynamic Hand Gestures from 3D Motion Data Using LSTM and CNN Architectures. In *Proceeding of the 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1130–1133, Cancun, Mexico, December 2017.

[63] Sergio Escalera, Vassilis Athitsos, and Isabelle Guyon. Challenges in Multi-modal Gesture Recognition. In *Gesture Recognition*, pages 1–60. 2017.

[64] Sidhant Gupta, Daniel Morris, Shwetak Patel, and Desney Tan. SoundWavee: Using the Doppler Effect to Sense Gestures. In *Proceedings of the ACM Annual Conference on Human Factors in Computing Systems (CHI)*, pages 1911–1914, Austin, Texas, USA, May 2012.

[65] Jun Haeng Lee, Tobi Delbruck, Michael Pfeiffer, Paul K. J. Park, Chang-Woo Shin, Hyunsurk Ryu, and Byung Chang Kang. Real-Time Gesture Interface Based on Event-Driven Processing From Stereo Silicon Retinas. *IEEE Transactions on Neural Networks and Learning Systems*, 25(12):2250–2263, December 2014.

[66] Christiane Attig, Nadine Rauh, Thomas Franke, and Josef F. Krems. System latency guidelines then and now – Is zero latency really considered necessary? In *Proceedings of the International Conference on Engineering Psychology and Cognitive Ergonomics (EPCE)*, pages 3–14, Vancouver, Canda, July 2017.

[67] Micael Carreira, Karine Lan Hing Ting, Petra Csobanka, and Daniel Gonçalves. Evaluation of in-air hand gestures interaction for older people. *Universal Access in the Information Society*, 16(3):561–580, 2017.

[68] Kwangtaek Kim, Joongrock Kim, Jaesung Choi, Junghyun Kim, and Sangyoun Lee. Depth camera-based 3d hand gesture controls with immersive tactile feedback for natural mid-air gesture interactions. *Sensors*, 15(1):1022–1046, 2015.

[69] Huidong Bai, Lei Gao, Jihad El-Sana, and Mark Billinghurst. Free-hand interaction for handheld augmented reality using an rgb-depth camera. In *Proceedings of the SIGGRAPH Asia Symposium on Mobile Graphics and Interactive Applications (MGIA)*, pages 1–4, Hong Kong, China, November 2013.

[70] Kyeong Beom Park and Jae Yeol Lee. New design and comparative analysis of smartwatch metaphor-based hand gestures for 3D navigation in mobile virtual reality. *Multimedia Tools and Applications*, 78(5):6211–6231, March 2019.

[71] Jamie A Ward, Paul Lukowicz, and Hans W Gellersen. Performance metrics for activity recognition. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(1):1–23, 2011.

[72] Jani Mäntyjärvi, Juha Kela, Panu Korpipää, and Sanna Kallio. Enabling fast and effortless customisation in accelerometer based gesture interaction. In *Proceedings of the 3rd International Conference on Mobile and Ubiquitous Multimedia (MUM)*, pages 25–31, New York, USA, October 2004.

[73] Glebys Gonzalez, Naveen Madapana, Rahul Taneja, Lingsong Zhang, Richard Rodgers, and Juan Pablo Wachs. Looking Beyond the Gesture: Vocabulary Acceptability Criteria for Gesture Elicitation Studies. In *Proceedings of the 62th Human Factors and Ergonomics Society Annual Meeting (HFES)*, volume 62, pages 997–1001, Philadelphia, Pennsylvania, USA, October 2018.

[74] Jakub Galka, Mariusz Masior, Mateusz Zaborski, and Katarzyna Barczewska. Inertial Motion Sensing Glove for Sign Language Gesture Acquisition and Recognition. *IEEE Sensors Journal*, 16(16):6310–6316, August 2016.

[75] M. Yeasin and S. Chaudhuri. Visual understanding of dynamic hand gestures. *Pattern Recognition*, 33(11):1805–1817, November 2000.

[76] Hessam Jahani and Manolya Kavakli. Exploring a user-defined gesture vocabulary for descriptive mid-air interactions. *Cognition, Technology and Work*, 20(1):11–22, February 2018.

[77] Johanna Höysniemi, Perttu Hämäläinen, Laura Turkki, and Teppo Rouvi. Children's intuitive gestures in vision-based action games. *Communications of the ACM*, 48(1):44, January 2005.

[78] Jungsoo Kim, Jiasheng He, Kent Lyons, and Thad Starner. The Gesture Watch: A Wireless Contact-free Gesture based Wrist Interface. In *Proceedings 11th IEEE International Symposium on Wearable Computers (ISWC)*, pages 1–8, Bosto, Massachusetts, USA, October 2007.

[79] Danilo Avola, Marco Bernardi, Luigi Cinque, Gian Luca Foresti, and Cristiano Massaroni. Exploiting Recurrent Neural Networks and Leap Motion Controller for the Recognition of Sign Language and Semaphoric Hand Gestures. *IEEE Transactions on Multimedia*, 21(1):234–245, January 2019.

[80] Thad Starner, Jake Auxier, Daniel Ashbrook, and Maribeth Gandy. The gesture pendant: a self-illuminating, wearable, infrared computer vision system for home automation control and medical monitoring. In *Proceedings of the 4th International Symposium on Wearable Computers (ISWC)*, pages 87–94, Atlanta, Georgia, USA, October 2000.

[81] Roberto Cipolla and Nicholas J. Hollinghurst. Human-robot interface by pointing with uncalibrated stereo vision. *Image and Vision Computing*, 14(3):171–178, April 1996.

[82] Mu-Chun Su. A fuzzy rule-based approach to spatio-temporal hand gesture recognition. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 30(2):276–281, 2000.

[83] Aditya Ramamoorthy, Namrata Vaswani, Santanu Chaudhury, and Subhashis Banerjee. Recognition of dynamic hand gestures. *Pattern Recognition*, 36(9):2069–2081, September 2003.

[84] Kai Nickel and Rainer Stiefelhagen. Visual recognition of pointing gestures for human–robot interaction. *Image and Vision Computing*, 25(12):1875–1884, December 2007.

[85] Gonzalo Bailador, Daniel Roggen, Gerhard Tröster, and Gracián Triviño. Real time gesture recognition using continuous time recurrent neural networks. In *Proceedings of the 2nd ICST International Conference on Body Area Networks (BodyNets)*, Florence, Italy, Jun 2007.

[86] Thomas Schlömer, Benjamin Poppinga, Niels Henze, and Susanne Boll. Gesture recognition with a Wii controller. In *Proceedings of the 2nd International Conference on Tangible and Embedded Interaction (TEI)*, pages 11–14, Bonn, Germany, February 2008.

[87] Jiayang Liu, Lin Zhong, Jehan Wickramasuriya, and Venu Vasudevan. uWave: Accelerometer-based personalized gesture recognition and its applications. *Pervasive and Mobile Computing*, 5(6):657–675, December 2009.

[88] Ahmad Akl and Shahrokh Valaee. Accelerometer-based gesture recognition via dynamic-time warping, affinity propagation, & compressive sensing. In *Proceedings of the 35th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2270–2273, Dallas, Texas, USA, March 2010.

[89] Tao Ni, Doug A Bowman, Chris North, and Ryan P McMahan. Design and evaluation of freehand menu selection interfaces using tilt and pinch gestures. *International Journal of Human-Computer Studies*, 69(9):551–562, 2011.

[90] Xu Zhang, Xiang Chen, Yun Li, Vuokko Lantz, Kongqiao Wang, and Jihai Yang. A framework for hand gesture recognition based on accelerometer and emg sensors. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 41(6):1064–1076, 2011.

[91] Mingyu Chen, Ghassan AlRegib, and Biing-Hwang Juang. A new 6D motion gesture database and the benchmark results of feature-based statistical recognition. In *Proceedings of the IEEE International Conference on Emerging Signal Processing Applications (ESPA)*, pages 131–134, Las Vegas, Nevada, USA, January 2012.

[92] Mridul Khan, Sheikh Iqbal Ahamed, Miftahur Rahman, and Ji-Jiang Yang. Gesthaar: An accelerometer-based gesture recognition method and its application in NUI driven pervasive healthcare. In *Proceedings of the IEEE International Conference on Emerging Signal Processing Applications (ESPA)*, pages 163–166, Las Vegas, Nevada, USA, January 2012. IEEE.

[93] Guilherme Cesar Soares Ruppert, Leonardo Oliveira Reis, Paulo Henrique Junqueira Amorim, Thiago Franco de Moraes, and Jorge Vicente Lopes da Silva. Touchless gesture user interface for interactive image visualization in urological surgery. *World journal of urology*, 30(5):687–691, 2012.

[94] Lorenzo Porzi, Stefano Messelodi, Carla Mara Modena, and Elisa Ricci. A smart watch-based gesture recognition system for assisting people with visual impairments. In *Proceedings of the 3rd ACM international workshop on Interactive multimedia on mobile & portable devices (IMMPD)*, pages 19–24, Barcelona, Spain, October 2013.

[95] Myeong-Chun Lee and Sung-Bae Cho. A Recurrent Neural Network with Non-gesture Rejection Model for Recognizing Gestures with Smartphone Sensors. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pages 40–46. 2013.

[96] Sundar Murugappan, Hairong Liu, Karthik Ramani, et al. Shape-it-up: Hand gesture based creative expression of 3d shapes using intelligent generalized cylinders. *Computer-Aided Design*, 45(2):277–287, 2013.

[97] Shengli Zhou, Fei Fei, Guanglie Zhang, John D. Mai, Yunhui Liu, Jay Y. J. Liou, and Wen J. Li. 2D Human Gesture Tracking and Recognition by the Fusion of MEMS Inertial and Vision Sensors. *IEEE Sensors Journal*, 14(4):1160–1170, April 2014.

[98] Zhiyuan Lu, Xiang Chen, Qiang Li, Xu Zhang, and Ping Zhou. A Hand Gesture Recognition Framework and Wearable Gesture-Based Interaction Prototype for Mobile Devices. *IEEE Transactions on Human-Machine Systems*, 44(2):293–299, April 2014.

[99] Kui Liu, Chen Chen, Roozbeh Jafari, and Nasser Kehtarnavaz. Fusion of Inertial and Depth Sensor Data for Robust Hand Gesture Recognition. *IEEE Sensors Journal*, 14(6):1898–1903, June 2014.

[100] Liang Yin, Mingzhi Dong, Ying Duan, Weihong Deng, Kaili Zhao, and Jun Guo. A high-performance training-free approach for hand gesture recognition with accelerometer. *Multimedia Tools and Applications*, 72(1):843–864, September 2014.

[101] Eshed Ohn-Bar and Mohan Manubhai Trivedi. Hand Gesture Recognition in Real Time for Automotive Interfaces: A Multimodal Vision-Based Approach and Evaluations. *IEEE Transactions on Intelligent Transportation Systems*, 15(6):2368–2377, December 2014.

[102] Stefan Duffner, Samuel Berlemont, Gregoire Lefebvre, and Christophe Garcia. 3D gesture classification with convolutional neural networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5432–5436, Florence, Italy, May 2014. IEEE.

[103] Guanglong Du and Ping Zhang. A markerless human–robot interface using particle filter and kalman filter for dual robots. *IEEE Transactions on Industrial Electronics*, 62(4):2257–2264, 2014.

[104] Baptiste Caramiaux, Nicola Montecchio, Atau Tanaka, and Frederic Bevilacqua. Adaptive gesture recognition with variation estimation for interactive systems. *ACM Transactions on Interactive Intelligent Systems*, 4(4), January 2015.

[105] Gorka Marques and Koldo Basterretxea. Efficient Algorithms for Accelerometer-Based Wearable Hand Gesture Recognition Systems. In *Proceedings of the IEEE 13th International Conference on Embedded and Ubiquitous Computing (EUC)*, pages 132–139, Porto, Portugal, October 2015.

[106] Yu-Liang Hsu, Cheng-Ling Chu, Yi-Ju Tsai, and Jeen-Shing Wang. An Inertial Pen With Dynamic Time Warping Recognizer for Handwriting and Gesture Recognition. *IEEE Sensors Journal*, 15(1):154–163, January 2015.

[107] Marcus Georgi, Christoph Amma, and Tanja Schultz. Recognizing Hand and Finger Gestures with IMU based Motion and EMG based Muscle Activity Sensing. In *Proceedings of the International Conference on Bio-inspired Systems and Signal Processing (BIOSIGNALS)*, pages 99–108, Lisbon, Portugal, January 2015. SCITEPRESS - Science and and Technology Publications.

[108] Chong Wang, Zhong Liu, and Shing-Chow Chan. Superpixel-Based Hand Gesture Recognition With Kinect Depth Camera. *IEEE Transactions on Multimedia*, 17(1):29–39, January 2015.

[109] Karthik Ramani et al. A gesture-free geometric approach for mid-air expression of design intent in 3d virtual pottery. *Computer-Aided Design*, 69:11–24, 2015.

[110] Tzuu-Hseng S Li, Min-Chi Kao, and Ping-Huan Kuo. Recognition system for home-service-related sign language using entropy-based *k*-means algorithm and abc-based hmm. *IEEE transactions on systems, man, and Cybernetics: systems*, 46(1):150–162, 2015.

[111] Bashar I Ahmad, James K Murphy, Patrick M Langdon, Simon J Godsill, Robert Hardy, and Lee Skrypchuk. Intent inference for hand pointing gesture-based interactions in vehicles. *IEEE transactions on cybernetics*, 46(4):878–889, 2015.

[112] Danial Moazen, Seyed A Sajjadi, and Ani Nahapetian. AirDraw: Leveraging smart watch motion sensors for mobile human computer interactions. In *Proceedings of the 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pages 442–446, January 2016.

[113] Feng Hong, Shujuan You, Meiyu Wei, Yongtuo Zhang, and Zhongwen Guo. MGRA: Motion Gesture Recognition via Accelerometer. *Sensors*, 16(4):530, April 2016.

[114] Hongyi Wen, Julian Ramos Rojas, and Anind K. Dey. Serendipity: Finger Gesture Recognition using an Off-the-Shelf Smartwatch. In *Proceedings of the ACM Annual Conference on Human Factors in Computing Systems (CHI)*, pages 3847–3851, San Jose, California, USA, May 2016. ACM Press.

[115] Renqiang Xie and Juncheng Cao. Accelerometer-Based Hand Gesture Recognition by Neural Network and Similarity Matching. *IEEE Sensors Journal*, 16(11):4537–4545, June 2016.

[116] Hari Prabhat Gupta, Haresh S. Chudgar, Siddhartha Mukherjee, Tanima Dutta, and Kulwant Sharma. A Continuous Hand Gestures Recognition Technique for Human-Machine Interaction Using Accelerometer and Gyroscope Sensors. *IEEE Sensors Journal*, 16(16):6425–6432, August 2016.

[117] Shalini Gupta, Pavlo Molchanov, Xiaodong Yang, Kihwan Kim, Stephen Tyree, and Jan Kautz. Towards selecting robust hand gestures for automotive interfaces. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, pages 1350–1357, Gothenburg, Sweden, June 2016.

[118] Pavlo Molchanov, Xiaodong Yang, Shalini Gupta, Kihwan Kim, Stephen Tyree, and Jan Kautz. Online Detection and Classification of Dynamic Hand Gestures with Recurrent 3D Convolutional Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4207–4215, Las Vegas, Nevada, USA, June 2016.

[119] Yimin Zhou, Guolai Jiang, and Yaorong Lin. A novel finger and hand pose estimation technique for real-time hand gesture recognition. *Pattern Recognition*, 49:102–114, 2016.

[120] Aashni Haria, Archanasri Subramanian, Nivedhitha Asokkumar, Shristi Poddar, and Jyothi S. Nayak. Hand Gesture Recognition for Human Computer Interaction. *Procedia Computer Science*, 115:367–374, 2017.

[121] Bo-Ra Shin, Heui-Su Son, Seok-Pil Lee, and Hyuk Soo Han. A gesture recognition system using a flexible epidermal tactile sensor based on artificial neural network. In *Proceedings of the IEEE International Conference on Robotics and Automation Sciences (ICRAS)*, pages 195–198, Hong Kong, China, August 2017.

[122] Nuno Mendes, João Ferrer, João Vitorino, Mohammad Safeea, and Pedro Neto. Human Behavior and Hand Gesture Classification for Smart Human-robot Interaction. *Procedia Manufacturing*, 11:91–98, 2017.

[123] Oyebade K. Oyedotun and Adnan Khashman. Deep learning in vision-based static hand gesture recognition. *Neural Computing and Applications*, 28(12):3941–3951, December 2017.

[124] Peijun Bao, Ana I. Maqueda, Carlos R. Del-Blanco, and Narciso García. Tiny hand gesture recognition without localization via a deep convolutional network. *IEEE Transactions on Consumer Electronics*, 63(3):251–257, August 2017.

[125] Hui Liang, Junsong Yuan, Jun Lee, Liuhao Ge, and Daniel Thalmann. Hough forest with optimized leaves for global hand pose estimation with arbitrary postures. *IEEE Transactions on Cybernetics*, 49(2):527–541, 2017.

[126] Ji-Hae Kim, Gwang-Soo Hong, Byung-Gyu Kim, and Debi P. Dogra. deepGesture: Deep learning-based gesture recognition scheme using motion sensors. *Displays*, 55:38–45, December 2018.

[127] Wei Zeng, Cong Wang, and Qinghui Wang. Hand gesture recognition using Leap Motion via deterministic learning. *Multimedia Tools and Applications*, 77(21):28185–28206, November 2018.

[128] Zhongxu Hu, Youmin Hu, Jie Liu, Bo Wu, Dongmin Han, and Thomas Kurfess. 3D separable convolutional neural network for dynamic hand gesture recognition. *Neurocomputing*, 318:151–161, November 2018.

[129] Chunyong Ma, Yu Zhang, Anni Wang, Yuan Wang, and Ge Chen. Traffic command gesture recognition for virtual urban scenes based on a spatiotemporal convolution neural network. *ISPRS International Journal of Geo-Information*, 7(1), January 2018.

[130] Minwoo Kim, Jaechan Cho, Seongjoo Lee, and Yunho Jung. IMU Sensor-Based Hand Gesture Recognition for Human-Machine Interfaces. *Sensors*, 19(18):3827, September 2019.

[131] Jinmiao Huang, Prakhar Jaiswal, and Rahul Rai. Gesture-based system for next generation natural and intuitive interfaces. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 33(1):54–68, February 2019.

[132] Lin Feng, Youchen Du, Shenglan Liu, Li Xu, Jie Wu, and Hong Qiao. Hand Gesture Recognition with Leap Motion. In *Advances in Intelligent Systems and Computing*, volume 880, pages 46–54. 2019.

[133] Zhang, Yang, Qian, and Zhang. Real-Time Surface EMG Pattern Recognition for Hand Gestures Based on an Artificial Neural Network. *Sensors*, 19(14):3170, July 2019.

[134] Marco A.F. Pimentel, David A. Clifton, Lei Clifton, and Lionel Tarassenko. A review of novelty detection. *Signal Processing*, 99:215–249, 2014.

[135] Michael Nielsen, Thomas B. Moeslund, Moritz Störring, and Erik Granum. Gesture Interfaces. In *HCI Beyond the GUI*, pages 75–106. Elsevier, 2008.

[136] Adil Mehmood Khan, Muhammad Hameed Siddiqi, and Seok-Won Lee. Exploratory data analysis of acceleration signals to select light-weight and accurate features for real-time activity recognition on smartphones. *Sensors*, 13(10):13099–13122, 2013.

[137] Barbara Bruno, Fulvio Mastrogiovanni, Antonio Sgorbissa, Tullio Vernazza, and Renato Zaccaria. Analysis of human behavior recognition algorithms based on acceleration data. In *Proceeding of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1602–1607, Karlsruhe, Germany, May 2013.

[138] Rupika Srivastava and Purnendu Sinha. Hand movements and gestures characterization using quaternion dynamic time warping technique. *IEEE Sensors Journal*, 16(5):1333–1341, 2016.

[139] Sungho Shin and Wonyong Sung. Dynamic hand gesture recognition for wearable devices with low complexity recurrent neural networks. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 2274–2277, Montreal, Canada, May 2016.

[140] Prasanta Chandra Mahalanobis. On the generalized distance in statistics. National Institute of Science of India, 1936.

[141] Nancy Chinchor. Muc-4 evaluation metrics. In *Proceedings of the 4th Conference on Message Understanding*, pages 22–29, McLean, Virginia, USA, June 1992.

[142] Allan Stisen, Henrik Blunck, Sourav Bhattacharya, Thor Siiger Prentow, Mikkel Baun Kjærgaard, Anind Dey, Tobias Sonne, and Mads Møller Jensen. Smart devices are different: Assessing and mitigatingmobile sensing heterogeneities for activity recognition. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, pages 127–140, Seoul, South Korea, November 2015.

[143] Gonzalo Bailador, Daniel Roggen, Gerhard Tröster, and Gracián Triviño. Real time gesture recognition using continuous time recurrent neural networks. In *Proceedings of the ICST International Conference on Body area networks (BODYNETS)*, page 15, Florence, Italy, June 2007.

[144] Gabriele Costante, Lorenzo Porzi, Oswald Lanz, Paolo Valigi, and Elisa Ricci. Personalizing a smartwatch-based gesture interface with transfer learning. In *Proceeding of the 22nd European Signal Processing Conference (EUSIPCO)*, pages 2530–2534, Lisbon, Portugal, September 2014.

[145] Daniel Conrad Halbert. *Programming by example*. PhD thesis, University of California, Berkeley, USA, 1984.

[146] Halit Bener Suay, Russell Toris, and Sonia Chernova. A practical comparison of three robot learning from demonstration algorithm. *International Journal of Social Robotics*, 4(4):319–330, 2012.

[147] Brenna Argall, Eric Sauser, and Aude Billard. Policy Adaptation through Tactile Correction. In *Proceedings of the Convention of the Society for the Study of Artificial Intelligence and Simulation of Behaviour (AISB)*, Leicester, United Kingdom, March 2010.

[148] Sonya Alexandrova, Maya Cakmak, Kaijen Hsiao, and Leila Takayama. Robot Programming by Demonstration with Interactive Action Visualizations. In *Proceedings of Robotics: Science and Systems (RSS)*, Berkeley, California, USA, July 2014.

[149] Chao Ping Tung and Avinash C Kak. Automatic Learning of Assembly Tasks Using a DataGlove System. In *Proceeding of the IEEE/RSJ Internationl Conference on Intelligent Robots and Systems (IROS)*, volume 1, Pittsburgh, Pennsylvania, USA, August 1995.

[150] Sing Bing Kang and Katsushi Ikeuchi. A robot system that observes and replicates grasping tasks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Boston, Massachusetts, USA, June 1995.

[151] Jens Lambrecht, Martin Kleinsorge, Martin Rosenstrauch, and Jörg Krüger. Spatial Programming for Industrial Robots through Task Demonstration. *International Journal of Advanced Robotic Systems*, 10(5):254, 2013.

[152] Sylvain Calinon, Florent Guenter, and Aude Billard. On Learning, Representing, and Generalizing a Task in a Humanoid Robot. *Neural Networks*, 37(2):286–298, 2007.

[153] Masato Ito, Kuniaki Noda, Yukiko Hoshino, and Jun Tani. Dynamic and interactive generation of object handling behaviors by a small humanoid robot using a dynamic neural network model. *Neural Networks*, 19(3):323–337, 2006.

[154] Tetsunari Inamura, Naoki Kojo, and Masayuki Inaba. Situation Recognition and Behavior Induction based on Geometric Symbol Representation of Multimodal Sensorimotor Patterns. In *Proceeding of the IEEE/RSJ Internationl Conference on Intelligent Robots and Systems (IROS)*, Beijing, China, October 2006.

[155] Petar Kormushev, Sylvain Calinon, and Darwin G Caldwell. Imitation Learning of Positional and Force Skills Demonstrated via Kinesthetic Teaching and Haptic Input. *Advanced Robotics*, 25(5):581–603, 2011.

[156] Daniele Massa, Massimo Callegari, and Cristina Cristalli. Manual guidance for industrial robot programming. *Industrial Robot: an International Journal*, 42(5):457–465, 2015.

[157] Sheng Liu and Haruhiko Asada. Teaching and learning of deburring robots using neural networks. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Atlanta, Georgia, USA, May 1993.

[158] Jie Yang, Yangsheng Xu, and Chiou S Chen. Hidden Markov Model Approach to Skill Learning and Its Application to Telerobotics. *IEEE transactions on Robotics and Automation*, 10(5):621–631, 1994.

[159] Rüdiger Dillmann, M Kaiser, and Ales Ude. Acquisition of Elementary Robot Skills from Human Demonstration. In *Proceedings of the International Symposium on Intelligent Robotics Systems (SIRS)*, Pisa, Italy, 1995.

[160] BB Chaudhuri. A new definition of neighborhood of a point in multi-dimensional space. *Pattern Recognition Letters*, 17(1):11–17, 1996.

[161] Geoffrey Biggs and Bruce MacDonald. A Survey of Robot Programming Systems. In *Proceedings of the Australasian Conference on Robotics and Automation (ACRA)*, Brisbane, Australia, 2003.

[162] E.K. Antonsson and R.W. Mann. The frequency content of gait. *Journal of Biomechanics*, 18(1):39–47, 1985.

[163] Diego R Faria, Ricardo Martins, Jorge Lobo, and Jorge Dias. Extracting data from human manipulation of objects towards improving autonomous robotic grasping. *Robotics and Autonomous Systems*, 60(3):396–410, 2012.

[164] Staffan Ekvall and Danica Kragic. Interactive grasp learning based on human demonstration. In *Proceeding of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 4, pages 3519–3524, New Orleans, Louisiana, USA, May 2004.

[165] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. In *Proceeding of the Robotics: Science and System Conference (RSS)*, Pittsburgh, Pennsylvania, USA, June 2017.